Learning a Hierarchical Intent Model for Next-Item Recommendation

NENGJUN ZHU and JIAN CAO, Department of Computer Science and Engineering, Shanghai Jiao Tong University
XINJIANG LU, Business Intelligence Lab, Baidu Research
HUI XIONG, Rutgers University

A session-based recommender system (SBRS) captures users' evolving behaviors and recommends the next item by profiling users in terms of items in a session. User intent and user preference are two factors affecting his (her) decisions. Specifically, the former narrows the selection scope to some item types, while the latter helps to compare items of the same type. Most SBRSs assume one arbitrary user intent dominates a session when making a recommendation. However, this oversimplifies the reality that a session may involve multiple types of items conforming to different intents. In current SBRSs, items conforming to different user intents have cross-interference in profiling users for whom only one user intent is considered. Explicitly identifying and differentiating items conforming to various user intents can address this issue and model rich contextual information of a session. To this end, we design a framework modeling user intent and preference explicitly, which empowers the two factors to play their distinctive roles. Accordingly, we propose a key-array memory network (KA-MemNN) with a hierarchical intent tree to model coarse-to-fine user intents. The two-layer weighting unit (TLWU) in KA-MemNN detects user intents and generates intent-specific user profiles. Furthermore, the hierarchical semantic component (HSC) integrates multiple sets of intent-specific user profiles along with different user intent distributions to model a multi-intent user profile. The experimental results on real-world datasets demonstrate the superiority of KA-MemNN over selected state-of-the-art methods.

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

Additional Key Words and Phrases: User modeling, representation learning, memory network, intent modeling, attention mechanism

ACM Reference format:
Nengjun Zhu, Jian Cao, Xinjiang Lu, and Hui Xiong. 2021. Learning a Hierarchical Intent Model for Next-Item Recommendation. *ACM Trans. Inf. Syst.* 40, 2, Article 38 (September 2021), 28 pages.
https://doi.org/10.1145/3473972

This work is a substantial revision of previously published work that has already appeared in the proceedings of the WSDM 2020.
This work is supported by National Key Research and Development Plan (No. 2018YFB1003800) and NSFC (No. 71531001).
Authors' addresses: N. Zhu and J. Cao (corresponding author), Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Dongchuan Road, Minhang District, Shanghai 200240, China; emails: {zhu_nj, cao-jian}@sjtu.edu.cn; X. Lu, Baidu Business Intelligence Lab, Baidu Research, 10 Xibeiwang East Road, Haidian District, Beijing, 100093, China; email: luxinjiang@baidu.com; H. Xiong, Rutgers University, 1 Washington Park, Newark, New Jersey, 07102, USA; email: hxiong@rutgers.edu.

1046-8188/2021/09-ART38 $15.00
https://doi.org/10.1145/3473972

1 INTRODUCTION

Recommender systems (**RSs**) can help users find their preferred items from a vast number of choices, thus enhancing user experience. Moreover, they help e-commerce platforms to increase user engagement and derive new business value. At the same time, user behavior data (e.g., click, purchase, and check-in) is growing exponentially and has been successively collected by platforms as a knowledge source to be fed into RSs. For instance, 62 million user trips were accumulated in July 2016 by Uber, and more than 10 billion check-ins were generated by over 50 million users at Foursquare [42]. As a result, SBRSs, which can respond to users' evolving behaviors, have become a hot research topic in recent years [23, 39, 41, 44]. They profile users by accumulating and filtering representative items' features in a **session context** (**SC**) and then recommend the next item according to the similarity between items and user profiles.

There are two major types of factors affecting the selection of items in a session, and in turn, affecting user profiles: (1) *User intent factors* imply a user's psychological context in which the session is started at that time. Most users visit RSs with certain intents in mind and directly browse items of some types. Thus, user intent is a factor narrowing search scope to some item types. (2) *User preference factors* affect the acceptance degree of a user to an item. Users have their potential criteria for comparing items and prefer ones whose attributes, e.g., brand and price, meet their requirements according to compensatory or non-compensatory rules [26]. In summary, for a session, user intent factors decide which item types are needed, while user preference factors determine the selections of items of the same type. Taking the case illustrated in Figure 1 as an example, a hostess planned a family party (i.e., a coarse-grained intent). She thus should prepare the dinner and decorate the house (i.e., two fine-grained intents). Accordingly, she browsed items corresponding to party supplies (i.e., a coarse-grained item category), including dinner food and decoration (i.e., two fine-grained item categories), in an e-shopping application and added a wine, a lobster, onions, balloons, and a rose (i.e., specific products) into the cart. Finally, she ended this shopping session by picking up a piece of bread for breakfast with the intent to save freight costs. In this case, items in the cart are all her prefers by comparing with non-selected ones conforming to the same intents. But there is no iPhone X in her cart could be better attributed to the intent factor rather than the preference factor.

Most existing SBRSs [12, 35, 42] do not distinguish between user intent and user preference. However, mixing intent with preference without identifying and differentiating SC for various user intents over a session may impair the algorithm's performance for two reasons: (1) Items conforming to different user intents have cross-interference in profiling users for whom one intent is considered. As shown in Figure 1, turquoise balloons are helpful to model the hostess's decoration preference but contribute less, even negatively, to model her food preference. (2) Typical SBRSs, including **recurrent neural networks** (**RNN**)-based and attention-based ones, are easy to fall into traps that recommending the next item according to only one arbitrary user intent [36]. In RNN-based approaches, the contextual items far from the target item will be overwhelmed by the near ones due to memory decay over time, and thus the most recent intent indicated by the nearest items will mostly take over the recommendation. Although attention-based approaches can relieve negative time influence, they tend to assign salient weights on very few significant items, making the intent of a session dominated by these few items [14]. Looking back to our example, these two types of approaches have a high probability to repeatedly recommend breakfast and dinner food, respectively. But, the reality is, the bread is just an add-on; instead, decoration products are needed more. Thus, the models fail to balance the effects of non-dominated user intents.

Currently, even though a small number of methods [36, 49] attempt to assign items in a session to build intent-specific user preference representations, they cannot identify users' high-level intents

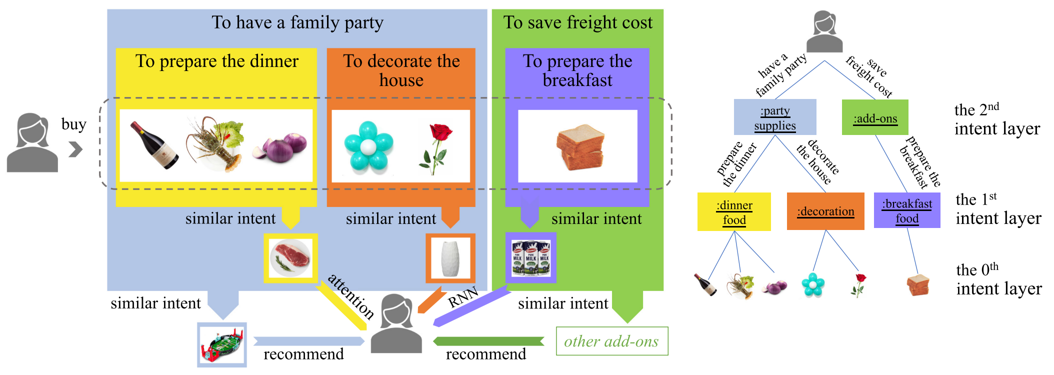


Fig. 1. The left subfigure is an illustration of different SBRSs to model a multi-intent session, where rectangular boxes with different colors manifest various user intents and the arrows with different colors show the very highly recommended items for their corresponding user intents. The right subfigure reveals the relationship between user intents and item categories, where a phrase that begins with a colon indicates the type/category of items conforming to the user intents above them.

since they often flatten the structure of SC so as to the hierarchical relationships between user intents are neglected. Thus, the recommended items would be very similar to the observed items conforming to a shallow intent, e.g., the first intent layer in Figure 1. The items, e.g., the board game for a party, which are also related according to a higher-level user intent, e.g., the second intent layer, have less chance of being recommended to the hostess in Figure 1. In [36], universal and abstract user intents are modeled for all sessions. It does not conform to the fact that users have various intents over different sessions.

To this end, we propose a framework to model user intent and preference explicitly. Unlike previous SBRSs [6, 25, 36, 45], which make user intent representation a mere figurehead and user preference representation an overladen mixture, our framework empowers user intent and preference factors to play their distinctive roles in a multi-factor modeling process, followed by a reasonable integration.

(1) *User intent factor modeling:* We assume multiple intent units (called intent for short), e.g., an item' category is a kind of intent units, an additional embedding space for representing different intents, and there is a multi-level intent tree forming the hierarchical semantic relationship between intents. For a new session, the involving intents (may with different granularity) and the SC for an intent is automatically identified according to a map (**soft router (SR)**) between observed items and intents. Then, we can explicitly analyze and calibrate the distribution of user intents over a session based on intent embeddings.
(2) *User preference factor modeling*: In addition to intent embeddings, there also exists a conventional item embedding space. Following the idea behind most SBRSs that treat a selective accumulation of item embeddings as user preference embedding (called user embedding for short) to profile users, we also learn a user embedding by linearly combining some observed items' embeddings. The difference is, we manipulate items within the scope of an SC related to an intent rather than scan all items at a time. We thus can learn multiple precise and intent-specific user embeddings over a session.
(3) *Factor integration*: As such, user intent factor modeling can help restore a user's psychological context and assign items for accurate user preference modeling. By integrating multiple sets of intent-specific user embeddings along with different user intent distributions, we finally profile a multi-intent user embedding to fit real intent-aware recommendations.

Specifically, we propose **Key-Array Memory Neural Network** (**KA-MemNN**), aiming at more concentratedly capturing intent-driven user behavior. To enhance model flexibility, KA-MemNN configures the user intent as a latent contextual variable rather than a fixed item category like in its last version [49], though it can handle both settings. After session partitioning based on a SR, the **two-layer weighting unit** (**TLWU**) in KA-MemNN first generates multiple intent-specific user embeddings using items' adaptive weights and then aggregates these embeddings based on their intent's distribution on the same intent layer. Thus, one TLWU can profile one intent layer-specific user embedding (called layer-specific embedding for short). Multiple TLWUs can be applied to different layers, respectively, and thus produce multiple layer-specific embeddings. Then, the **hierarchical semantic component** (**HSC**) in KA-MemNN builds a multi-intent user embedding by integrating all the layer-specific embeddings. Finally, to track more historical information based on **memory neural networks** (**MemNNs**), we adopt two HSCs and apply them to the short-term and the merged long-term sessions, respectively. As a result, a hybrid user embedding is generated to evaluate items for the next-item recommendations.

Our contributions are summarized as follows:

(1) We propose a learning framework for intent-sensitive recommendations to capture rich intent-related contextual information. Accordingly, we design an MemNN, i.e., KA-MemNN, to achieve this.

(2) We perform experiments on real-world datasets, and the results show that our model consistently outperforms state-of-the-art methods in terms of Recall and **mean reciprocal rank (MRR)** metrics. The ablation experiments, parameter analysis, and visualization prove that our model's design is practical and works as claimed.

The remainder of this article is organized as follows: We first introduce the related work in Section 2. In Section 3, we give the definitions and notations, describe KA-MemNN in detail, and introduce the technical relevance between KA-MemNN and its last version. The experimental evaluation is detailed in Section 4, followed by the conclusions and future work in Section 5.

2 RELATED WORK

Our KA-MemNN addresses next-item recommendation problems, and it conducts **dynamic representation learning** (**DRL**) for users through an MemNN. Therefore, our approach is related to session-based RSs, e.g., sequential modeling and DRL, and MemNNs. Furthermore, the main motivation behind KA-MemNN is user intent modeling. The intent can be embodied using item categories to some extent. Thus, we also introduce some RSs with the influence of intents or item categories being considered. Lastly, we compare KA-MemNN with its original version.

2.1 Session-Based Recommendation

SBRS spits user behaviors into multiple sessions according to timestamps associated with each behavior record. It can conduct sequential modeling or DRL over sessions according to various strategies and motivations.

Sequential modeling. There are two major types of sequential RSs, namely **Markov Chain** (**MC**) and RNN-based RSs. MCs have become classic tools to handle users' sequentially generated behaviors through sequential pattern mining [31, 40] and transition modeling [9, 30, 47]. For instance, based on the insight that frequent patterns can be utilized to predict the next items users will access, the work in [40] emphasized personalities in their models by discovering user-specific frequent patterns. In contrast, the work in [30] proposed FPMC, which directly utilizes MF to capture users' general tastes and combines MF with a first-order MC to predict the next item based on the recent items by learning a transition graph over the items. Despite the effectiveness of these methods, there is a homogeneous hypothesis behind MCs. It is a strong assumption and might impair the recommendation performance. Recently, many researchers [6, 11, 18, 21, 38] have begun to embrace RNN, including **gated recurrent unit** (**GRU**) and **long short-term memory** (**LSTM**), in RSs since it has the power not only to highlight the short-term status but also to track the long-term information. Like MC-based RSs, RNN-based RSs also treat a session as a sequence and usually limit user interactions within a short time frame. They thus can input each item in a session into RNN successively. For example, GRU4Rec [11] applies GRU-based RNN to SBRS and achieves a significant improvement over traditional methods. KSR [18] and SSRM [6] take GRU as one core component in their architectures to model users' sequential behaviors. Then, the outputs of GRU are further utilized or integrated by the other components. Besides, the work in [51] proposed a time-aware LSTM to consider sequential information and time interval information simultaneously. Some sequential RSs such as SASRec [19] don't directly use RNN, but they follow the idea behind RNNs and also consider a session as a sequence. These approaches are usually good at modeling the current short sequence but are much weakened in modeling long sequences due to memory decay over time.

Dynamic representation learning. Items in a session may not follow a rigidly internal order in many real scenarios, e.g., items in a shopping cart, and the behaviors outside the current session are also important. Thus, many researchers have begun to maintain the order of sessions but relax that of items inside a session. They treated each session as a set and left each items' importance to be learned automatically when generating users' representations. As a result,DRL has attracted immense attention recently. The RSs based on DRL consider items in a session collectively rather than successively. Furthermore, they combine users' previously and recently generated sessions using different contributions or strategies to learn a hybrid representation. For instance, the work in [12] and [35] both employed a two-layer structure to construct a hybrid representation over users and items. The authors of [42] argued that the weights of different components should not be fixed and thus proposed an attention-based **sequential hierarchical attention network** (**SHAN**). HyperRec [34] constructs a hypergraph for each time interval, which a **graph convolution network** (**GCN**) can be applied. Then, by considering the sequential hypergraphs, the model can output a hybrid representation for prediction. Our work follows this pipeline and conducts DRL for users, the main difference being that we creatively introduce an intent layer above items to enhance traditional weighting mechanisms.

2.2 Memory Neural Networks

MemNNs have been proven useful for a variety of document reading and question answering tasks, such as **end-end memory neural networks** (**EE-MemNNs**) [32] and **key-value memory neural networks** (**KV-MemNNs**) [28]. The memory component, i.e., the memory bank, of MemNNs can increase modeling capacity as well as generate a more informative representation of historical knowledge to track long-term dependencies. For instance, a **collaborative memory network** (**CMN**) [5] takes users who have co-click/co-visitation behaviors with the target user on a specific item as the neighborhoods. These neighborhoods are stored in internal memory to accumulate local neighborhood-based information. CMN is a non-session approach and learns the static representations of users, and thus, it is inadequate for users' dynamic preference modeling. In contrast, RUM [3] introduces a user memory matrix to maintain a user's recently visited items. User embeddings are formed according to the retrieved information. RUM adopts a simple first-in-first-out mechanism to maintain the latest interacted items in the user memory matrix. Thus, older information is forgotten gradually. Like CMN and RUM, the applications of MemNNs in RSs are usually based on user-item binary relations [3]. So far, limited work utilizes MemNNs with user (latent) intent being considered. To consider additional **knowledge base** (**KB**) information, KSR [18] integrates the GRU-based networks with KV-MemNN and stores attribute-level preference representations into memory banks. However, the key setting of KSR is different from ours since we don't utilize KB information.

Recently, TMRN [16], in which MemNNs have been first combined with taxonomy information, was proposed. TMRN can be viewed as one taking user intent into account because user intents are highly related to item categories. TMRN attempts to encode the hierarchical category semantics but without incorporating item information. Thus, it cannot fully exploit the link information from user-intent-item triadic relations. Furthermore, each hop in TMRN accumulates the information of one layer of item categories instead of a session, causing the approach to have a limited ability to acquire users' dynamical preference information.

2.3 Approaches with Intent Influence Considered

As mentioned, there are two types of approaches that incorporate the influence of intents into models. Some models consider latent user intents while the other methods embody user intents with other side information such as item categories. Nevertheless, both investigate user behavioral intents based on user behavioral data and learn additional latent intent representations. For instance, the work in [25] generated user intent/motivation representations by modeling users' various behaviors, e.g., click, collect, cart, and purchase. SSRM [6] and AttRec [45] both select the most relevant items in a session to form the representation of user intents. The relevance is determined by the MF-attention in SSRM and the self-attentive map in AttRec. However, these approaches usually assume the items inside a session are only associated with one implicit intent. In such a case, there is no significant difference between user intent and user preference. In contrast, MCPRN [36] learns multiple latent intent representations for one session. The effectiveness of this design has been proven in [36]. Unfortunately, the number of latent intents is decided in an arbitrary way. Moreover, intents are not structurally organized but treated equally.

Recently, some works [8, 37, 46] model user historical behaviors considering the influence of item category information. In particular, CHLF_I [33] deduces the effect of categorical information on items. But it is a non-session approach and neglects user-category relations. TMRN [16] attempts to encode the hierarchical category semantics. However, it only utilizes the structure of item categories and the representations of categories are not semantically linked to the representations of items. Thus, both CHLF_I and TMRN do not learn user-intent-item triadic relations sufficiently supposing item categories are used to represent user intents. Furthermore, unlike our model, neither of them can be applied to learning latent multi-level intents when the taxonomy information is unknown.

2.4 Differences from the Original Version

The original work [49] has been published in the proceedings of the WSDM 2020 and the old model is renamed to KA-MemNN-CA in this article to distinguish with the new model, i.e., KA-MemNN. KA-MemNN and KA-MemNN-CA both model user intent and preference in a learning framework. In KA-MemNN-CA, user intents must be embodied by item categories, causing it is not qualified for the situation when the taxonomy information of items is not available. To handle this problem, KA-MemNN is proposed based on KA-MemNN-CA. Specifically, in KA-MemNN, we utilize the representation of latent user intents to replace that of real-world item categories. By doing so, KA-MemNN can also deal with multi-level intents, which KA-MemNN-CA cannot do. In this perspective, KA-MemNN-CA can be viewed as a special case of KA-MemNN. Moreover, to better train the new model, we also define a novel loss function by considering the relations between different intents. This loss function helps shape the structure of the intent tree.

The organization of the new manuscript also varies from the old one. In this version, we describe KA-MemNN in detail and briefly introduce how to generate KA-MemNN-CA by modifying

Table 1. Descriptions of Mathematical Notations

Notation	Description
$\boldsymbol{h}_u \in \mathbb{R}^k$	The initial user embeddings, where k is the dimensionality of the vector.
$\boldsymbol{Z}, \boldsymbol{Z}^i \in \mathbb{R}^{n_{c_i} \times k}$	The intent-specific user embedding matrices, where i marks the layer of intents and n_{c_i} represents the number of intents in that layer.
$\boldsymbol{o}, \boldsymbol{o}^i \in \mathbb{R}^k$	The layer-specific user embeddings.
$\boldsymbol{f}, \boldsymbol{f}_L, \boldsymbol{f}_S \in \mathbb{R}^k$	The multi-intent user embeddings.
$\boldsymbol{e} \in \mathbb{R}^k$	The item embeddings.
$\boldsymbol{V} \in \mathbb{R}^{\lvert\mathcal{I}\rvert \times k}$	The item embedding matrix of all items, where $\lvert\mathcal{I}\rvert$ is the number of all items.
$\mathcal{S}, \mathcal{S}_L, \mathcal{S}_S$	The sessions (sets) of items, where L and S identify long and short-term.
$\boldsymbol{X}, \boldsymbol{X}_L, \boldsymbol{X}_S \in \mathbb{R}^{\lvert\mathcal{S}\rvert \times k}$	The session matrices, where $\lvert\mathcal{S}\rvert$ is the number of items in the session of $\boldsymbol{X}$, i.e., $\mathcal{S}$.
$\boldsymbol{C}, \boldsymbol{C}^i \in \mathbb{R}^{n_{c_i} \times k}$	The intent embedding matrices, where i marks the layer of intents and n_{c_i} represents the number of intents in that layer.
$\boldsymbol{M}, \boldsymbol{M}^i \in \mathbb{R}^{\lvert\mathcal{S}\rvert \times n_{c_i}}$	The item-intent map matrices, which record the relations between $\lvert\mathcal{S}\rvert$ items and n_{c_i} intents in the ith layer.

KA-MemNN. Also, in the experiment section, the various experiments are conducted to test the effectiveness of KA-MemNN. Whereas, KA-MemNN-CA is more like a baseline to be compared.

3 KEY-ARRAY MEMORY NEURAL NETWORK (KA-MEMNN)

In this section, we first formulate the next-item recommendation problem, and then introduce the details of our model, followed by a description of the model learning strategy.

3.1 Problem Formulation

Let the symbol $\mathcal{I}$ represent the set of all items and $\boldsymbol{e} \in \mathbb{R}^k$ be an item embedding, where k is the dimensionality. Let the symbol $\boldsymbol{V} = [\ldots, \boldsymbol{e}_i, \ldots]^\top, \forall i \in \mathcal{I}$ be all items' embedding matrix. According to timestamps of interactions, the visited items of a user can be split into multiple sets/sessions $\mathcal{S} \subseteq \mathcal{I}$. Furthermore, each one has a session matrix $\boldsymbol{X}$ related to its items as follows:

$$\mathcal{S} \overset{generate}{\longrightarrow} \boldsymbol{X} = [\ldots, \boldsymbol{e}_i, \ldots]^\top \in \mathbb{R}^{\lvert\mathcal{S}\rvert \times k}, \forall i \in \mathcal{S},$$

where $\lvert \cdot \rvert$ represents the number of elements in the set. For simplicity and following the settings in some previous RSs [42, 49], we further divide the sessions of each user into two parts, i.e., long- and short-term sessions, and utilize uppercase letters $\{L, S\}$ to identify them. More specifically, we let $\mathcal{S}_S$ (resp. $\boldsymbol{X}_S$) denote a user's current session (resp. the current session matrix) and $\mathcal{S}_L$ (resp. $\boldsymbol{X}_L$) denote the union set (resp. the matrix of the union set) of all sessions before the current session. $\mathcal{S}_S$ ($\boldsymbol{X}_S$) reflects a user's current status and $\mathcal{S}_L$ ($\boldsymbol{X}_L$) implies a user's historical referable behavior information. All vectors throughout the article are column ones and the main notations are listed in Table 1.

Formally, given a user and his (her) two item sessions $\mathcal{S}_L$ and $\mathcal{S}_S$, we aim to recommend the next items, which most likely belong to the current session $\mathcal{S}_S$.

3.2 Overview

We can accumulate relevant information from users' historical sessions incrementally to complete our task. Since MemNNs can track long-term dependencies by reading memories multiple times, following this pipeline, we design a KA-MemNN to conduct next-item recommendations.

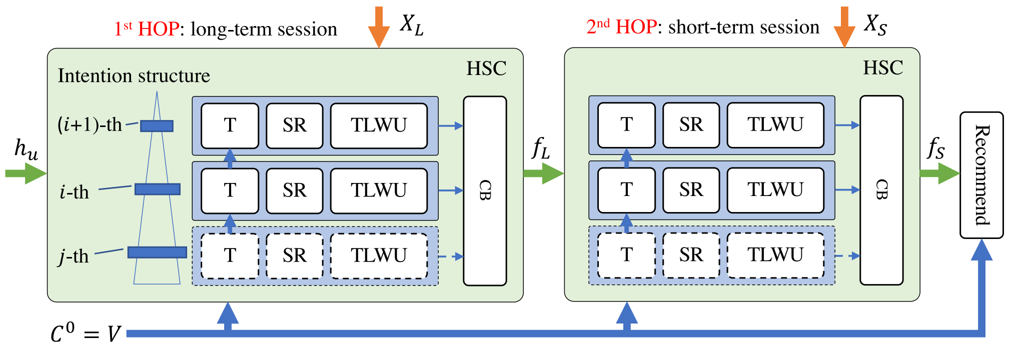


Fig. 2. The overview of the network architecture.

Our model shown as Figure 2 has two hops (each hop is also referred to as one iteration or one memory bank [28]), where one is for the long-term session, and the other is for the short-term session. Each hop is conducted by a HSC that is detailed in Section 3.6. This component is devised for modeling the influence from user intents varying from fine-grained to coarse-grained, i.e., from the bottom, e.g., the jth layer, to the top, e.g., the (i+1)th layer. HSC has a multi-layer network structure, which corresponds to different levels of intents. The unit CB, which is also seen in Section 3.6, is in charge of fusing the influences from different layers. In each layer, there are three vital units: T, SR (introduced in Section 3.4), and TLWU (introduced in Section 3.5). T is a transformer that converts intent embeddings in a lower layer to a higher one. SR is a soft router that maps items to different intents according to the linking strengths between item embeddings and intent embeddings in the corresponding layer. TLWU is a two-layer weighting unit that aggregates a session matrix to a layer-specific user embedding by considering intent-specific user embedding and user intent distribution.

Each hop has three inputs for HSC: an initial user embedding $\boldsymbol{h}_u$, a session matrix $\boldsymbol{X}$, and an initial intent embedding matrix $\boldsymbol{C}^0$ defined as the item embedding matrix $\boldsymbol{V}$. In our model, the output of the first hop, i.e., the long-term multi-intent user embedding $\boldsymbol{f}_L$, is a base of the second hop to replace its $\boldsymbol{h}_u$. In doing so, the model can track long-term dependencies. Then, through two hops of accumulation, we obtain the short-term multi-intent user embedding (called hybrid user embedding for short and detailed in Section 3.6) $\boldsymbol{f}_S$, which are utilized to make recommendations. Next, we introduce each part of the model in detail.

3.3 General Embedding Construction

In this article, we only utilize users' historical behavior data such as view, click, and purchase. Since the basic IDs of users and items (i.e., one-hot representations) have a very restricted representation capability, we first employ a fully connected layer with an embedding matrix $\boldsymbol{V} \in \mathbb{R}^{|\mathcal{I}|\times k}$ to construct a continuous low-dimensional embedding of items, which is similar to discrete word symbols in natural language processing [43]. We feed the fully connected layer network with one-hot representations of items rather than other side information as in [1, 15, 50], though our model also has an excellent capacity to deal with this. Then, the network outputs the corresponding embeddings $\boldsymbol{e}_i$ for item i, i.e., the ith row of $\boldsymbol{V}$. Although an initial user embedding $\boldsymbol{h}_u$ can be obtained in the same way, we generate $\boldsymbol{h}_u$ based on the mean of embeddings of all items in $\mathcal{S}_L$ as follows since this setting can speed up the model training process and make our model perform more stable:

$$\boldsymbol{h}_u = \phi(\boldsymbol{W}_u \cdot mean(\boldsymbol{X}_L) + \boldsymbol{b}_u),$$

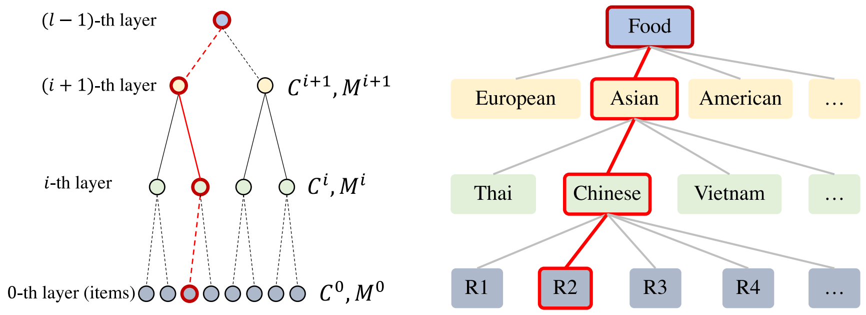


Fig. 3. The structure of a multi-level intent tree. For ease of understanding, item categories are directly treated as different user intents, e.g, the phrase "Chinese" here represents "the intent to eat Chinese food".

where $\boldsymbol{W}_u \in \mathbb{R}^{k \times k}$ and $\boldsymbol{b}_u \in \mathbb{R}^k$ are model parameters of a **multi-layer perceptron** (**MLP**) [7]. $\phi(\cdot)$ is an activation function and we utilize the RELU function to enhance the non-linear capability and filter out some trivial information [24].

The item embedding $\boldsymbol{e}$ is a static representation since it does not differentiate between users' new and old behaviors. In contrast, embedding $\boldsymbol{h}_u$ is a dynamic representation depicting users' drifting interests as long as $\mathcal{S}_L$ has been changed. In traditional non-session RSs, such as BPR [29] and CMN [5], a static representation is usually model for both users and items. This is a critical limitation, and we further address this problem by introducing the session-based TLWU and HSC, which will be discussed later.

3.4 Latent Multi-level Intent Tree

Assume that an item can match multiple intents, and an universal (user-independent) intent tree with l layers is shown in Figure 3. All items are located at the bottom and displayed as leaf nodes. For example, all restaurants are located in the 0th layer in the right subfigure of Figure 3. Furthermore, each layer i of the intent tree has an embedding matrix denoted as $\boldsymbol{C}^i \in \mathbb{R}^{n_{c_i} \times k}$, where n_{c_i} is the number of nodes (intents) in the ith layer and each row of $\boldsymbol{C}^i$ denotes an intent embedding. The information of the lower layers is merged into the higher layers, thus there exists a transmission relation between the embedding matrices of different layers, which can be defined as a transformer T as follows:

$$\begin{aligned} \boldsymbol{C}^0 &\overset{def}{=} \boldsymbol{V}, \\ \boldsymbol{C}^{i+1} &= \mathrm{T}_i(\phi(\boldsymbol{C}^i)), \end{aligned} \tag{1}$$

where $\boldsymbol{C}^{i+1} \in \mathbb{R}^{n_{c_{i+1}} \times k}$ and $\mathrm{T}_i(\cdot) : \mathbb{R}^{n_{c_i}} \mapsto \mathbb{R}^{n_{c_{i+1}}}$ is a fully connected NN. We set $n_{c_{i+1}} < n_{c_i}$ to shape a pyramid structure of the latent intent tree. Then, each item in a session can be assigned to different intents in this tree through a SR, which is based on the connection strength between its embedding and intent embeddings as follows:

$$\begin{aligned} \boldsymbol{M}^i &= \mathbb{I}(\boldsymbol{X}\boldsymbol{C}^{i\top}) \\ &= [\dots, \boldsymbol{m}^i_j, \dots]^\top, \forall j \in \mathcal{S} \wedge \boldsymbol{m}^i_j \in \{0,1\}^{n_{c_i}}, \end{aligned} \tag{2}$$

where $\mathbb{I}(\cdot)$ is an element-wise binary function to convert positive values to ones and non-positive values to zeros. The discrete matrix $\boldsymbol{M}$ of a session is an output of SR and $\boldsymbol{m}^i_j$ is a multiple-hot vector that maps item j to different intents in layer i. Accordingly, the cth column of $\boldsymbol{M}$, e.g., $\boldsymbol{M}_{:c}$, identifies which items in session $\mathcal{S}$ match the intent c.

Based on a session, different user intents can be highlighted from bottom up starting from the visited item shown as the red path of Figure 3. They have related semantic information with different granularities varying from fine- to coarse-grained. Thus, the highlighted nodes should have close representations. According to this observation and assumption, we add a constraint term and minimize the loss as follows:

$$\boldsymbol{M}^0\boldsymbol{C}^0 \overset{def}{=} \boldsymbol{X},$$
$$\mathcal{L}_{intent} = ln\sigma\left(\sum_{i=1}^{l-1} \| \boldsymbol{M}^i\boldsymbol{C}^i - \boldsymbol{M}^{i-1}\boldsymbol{C}^{i-1} \|_2\right), \quad (3)$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is a logistic function and $\| \cdot \|_2$ is an L2-norm operation. In this way, we make the branches under the same nodes have a similar intent embedding indirectly.

To simplify the problem, we assume the initial latent intent tree is a full N-ary tree, and the number of leaf nodes is equal to that of the items. Then, there is only a hyper-parameter, i.e., the number of layers l, deciding the outline of the latent tree. The specific structures, i.e., the edges between nodes, are auto-adjusted according to the SR matrix $\boldsymbol{M}$. Based on the hypothetical outline of the tree, we can calculate the number of nodes in each layer, i.e., n_{c_i}, as follows:

$$n_{c_i} = \begin{cases} base^{(l-1-i)}, & 0 < i \le l-1, \\ |\mathcal{I}|, & i = 0, \end{cases}$$
$$base = \lfloor \sqrt[(l-1)]{|\mathcal{I}|} \rfloor,$$

where the function $\lfloor\cdot\rfloor$ makes fractions round down and n_{c_i} is the $(l-1-i)$th power of $base$ when $0 < i \le l-1$.

3.5 Two-Layer Weighting Unit (TLWU)

The relationship between user intents and specific items has a hierarchical structure. Thus, we design a TLWU to model this relation based on a session $\mathcal{S}$, shown in Figure 4. The items' weights, i.e., $\boldsymbol{w}_A$, which are the linking strength between items and the target user, are viewed as determining factors to profile an intent-specific user embedding. The intent distribution, i.e., $\boldsymbol{w}_K$, which are the linking strength between user intents and the target user, differentiates and calibrates the importance of user intents. Most traditional approaches emphasize the different impacts from items and only assemble item weighting mechanism. In contrast, we introduce an additional intent adjuster to amplify or weaken the signal from items once again according to different user intents.

Specifically, the initial user embedding $\boldsymbol{h}_u$ is utilized to generate two sets of weights, i.e., item weights and intent weights. They are also identified as *array* weights $\boldsymbol{w}_A$ and *key* weights $\boldsymbol{w}_K$ in our KA-MemNNs, respectively. Following most RSs considering the influence from items [12, 27, 42], the $\boldsymbol{w}_A$ is determined as follows:

$$\boldsymbol{w}_A = softmax(\boldsymbol{X}\boldsymbol{h}_u), \quad (4)$$

where $\boldsymbol{w}_A$ is the Softmax result of the vector $\boldsymbol{X}\boldsymbol{h}_u$.

With an item-intent map $\boldsymbol{M}^i$ defined in Equation (2), the items can be aggregated to form an intent-specific user embedding matrix $\boldsymbol{Z}^i \in \mathbb{R}^{n_{c_i}\times k}$. The cth row of $\boldsymbol{Z}^i$ denoted as $\boldsymbol{Z}^i_{c:}$, represents the cth intent-specific user embedding and is calculated as follows:

$$\boldsymbol{Z}^i_{c:} = \boldsymbol{X}^\top(\boldsymbol{M}^i_{:c} \otimes \boldsymbol{w}_A),$$

where $\boldsymbol{M}^i_{:c}$ is the cth column of $\boldsymbol{M}^i$ and $\otimes$ operator is an element-wise multiplication. $\boldsymbol{Z}^i$ is utilized as *values* to be aggregated and $\boldsymbol{C}^i$ is utilized as a *key* to guide the probability in an attention

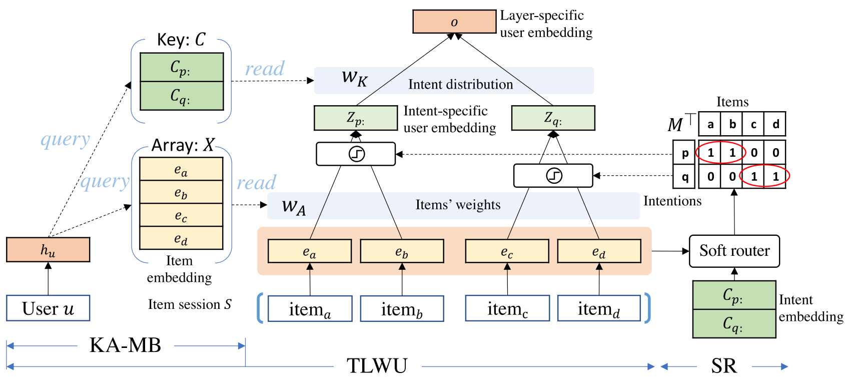


Fig. 4. TLWU and SR.

mechanism [2] or KV-MemNN [28]. Similar to the definition of $\boldsymbol{w}_A$ in Equation (4), we calculate $\boldsymbol{w}_K$ as follows:

$$\boldsymbol{w}_K^i = softmax(C^i \boldsymbol{h}_u).$$

Finally, we aggregate $\boldsymbol{Z}^i$ by a linear combination to produce a layer-specific user embedding $\boldsymbol{o}^i$ as follows:

$$\boldsymbol{o}^i = \boldsymbol{Z}^{i\top}\left(\mathbb{I}\left(\sum_j \boldsymbol{M}_{j:}^i\right) \otimes \boldsymbol{w}_K^i\right),$$

where the multi-hot vector $\mathbb{I}(\sum_j \boldsymbol{M}_{j:}^i)$ is a mask vector indicating, which user intents in the ith layer are involved.

TLWU can be viewed as an extension of KV-MemNNs [28]. For a standard KV-MemNN, assume we have a query vector $\boldsymbol{h}_u \in \mathbb{R}^k$, then the task is to reconstruct $\boldsymbol{h}_u$ to a more informative representation by retrieving n key-value pairs: $\boldsymbol{K} \in \mathbb{R}^{n\times k}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times k}$ from the memory bank.

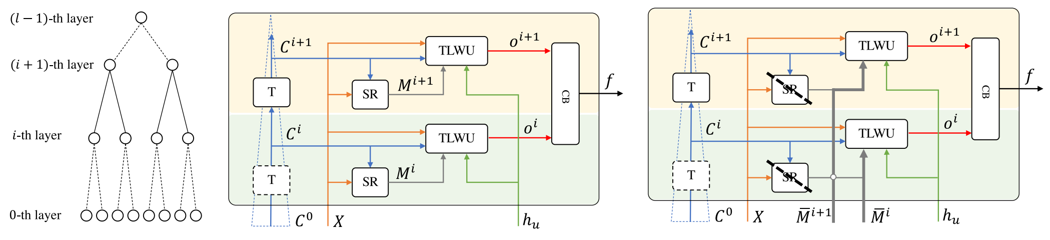
$$kv(\boldsymbol{h}_u, \boldsymbol{K}, \boldsymbol{V}|\omega) = \omega(\boldsymbol{K}\boldsymbol{h}_u)^\top \boldsymbol{V},$$

where $kv(\cdot)$ is a KV-MemeNN function, and $\omega(\boldsymbol{K}\boldsymbol{h}_u) \in \mathbb{R}^n$ is a weight vector, or affinity vector describing the similarity between the target *query* and the corpus. $\omega(\cdot)$ is usually an activation function like RELU and Softmax. Then, the $\boldsymbol{h}_u$ is reconstructed by a weighted sum of $\boldsymbol{V}$, where each *value* gets the weight from the similarity between its corresponding *key* and the target *query*. When the *key* and *value* become the same, i.e., $\boldsymbol{K} = \boldsymbol{V}$, the KV-MemNN degrades to the EE-MemNN [32].

Instead, our TLWU generalizes KV-MemNN by replacing the single *value* with a variable-length *array*, which records multiple *values* for each *key*, i.e., the single matrix $\boldsymbol{V}$ is replaced by two matrices $\boldsymbol{X}$ and $\boldsymbol{M}$. The *values* in the same *array* have the same property, which makes them have a shared *key*. For example, all *values* satisfy a common intent, i.e., their corresponding mask codes in $\boldsymbol{M}_{:c}$ are all ones. We refer to this data structure as having a key-array data type. Formally, given *query* $\boldsymbol{h}_u$, *key* matrix $\boldsymbol{C}$, *array* matrix $\boldsymbol{X}$, and *map* matrix $\boldsymbol{M}$, TLWU reconstructs a hybrid representation $\boldsymbol{o}$ of $\boldsymbol{h}_u$ as follows:

$$\boldsymbol{o} = \text{TLWU}(\boldsymbol{h}_u, \boldsymbol{C}, \boldsymbol{X}, \boldsymbol{M}), \tag{5}$$

where map matrix $\boldsymbol{M}$ can be calculated as Equation (2), or obtained from an additional taxonomy information of items. Then, the process of calculating $\boldsymbol{w}_K$ and $\boldsymbol{w}_A$ can be viewed as two *query* operations on our **key-array memory bank** (**KA-MB**), as shown as Figure 4.



(a) HSC (b) HSC with the SR removed

Fig. 5. HSC, where T is a transformer introduced in Equation (1), SR is a soft router defined in Equation (2), and CB is a weighted combination operation.

3.6 Hierarchical Semantic Component (HSC)

We have introduced how to apply one TLWU to one layer of user intents in the last subsection. However, an intent tree has multiple layers and each layer represents the different granularities as aforementioned. To fuse the impacts from all layers, we design a HSC based on multiple TLWUs shown in Figure 5(a).

For the ith layer in an intent tree, we have an intent embedding matrix $\boldsymbol{C}^i$ introduced in Equation (1) and a map matrix $\boldsymbol{M}^i$ introduced in Equation (2). Thus, we can apply TLWU to different layers as follows:

$$\boldsymbol{o}^i = \begin{cases} \text{TLWU}(\boldsymbol{h}_u, \boldsymbol{C}^i, \boldsymbol{X}, \boldsymbol{M}^i), & 0 < i \leq l-1, \\ \boldsymbol{X}^\top \boldsymbol{w}_A, & i = 0. \end{cases}$$

When $i = 0$, we only consider items' weights in TLWU to output $\boldsymbol{o}^i$. Then, we employ a one-layer fully connected NN $fc_H(\cdot) : \mathbb{R}^k \mapsto \mathbb{R}$ and a Softmax function to automatically adjust the importance of different layers.

$$\boldsymbol{w}_H = softmax([\ldots, fc_H(\boldsymbol{o}^i), \ldots]), i = 0, 1, 2, \ldots.$$

Finally, we fuse all layer-specific user embeddings to a multi-intent user embedding $\boldsymbol{f} \in \mathbb{R}^k$ based on a weighted combination CB as follows:

$$\boldsymbol{f} = [\ldots, \boldsymbol{o}^i, \ldots]\boldsymbol{w}_H.$$

Therefore, $\boldsymbol{f}$ is an output of HSC$(\cdot)$ function that is summarized as follows:

$$\begin{aligned} \boldsymbol{f} &= \text{HSC}(\boldsymbol{h}_u, \boldsymbol{C}^0, \boldsymbol{X}) \\ &= \text{HSC}(\boldsymbol{h}_u, \boldsymbol{V}, \boldsymbol{X}). \end{aligned}$$

From the pipeline of attention mechanisms [13, 45], HSC$(\cdot)$ can be viewed as a superior attention model to learn elaborate and robust representations with limited knowledge of the whole context.

Long and short-term modeling: The aforementioned HSC is a session-based component. Since we split a user's sessions into two parts: $\mathcal{S}_L \overset{generate}{\longrightarrow} \boldsymbol{X}_L$ and $\mathcal{S}_S \overset{generate}{\longrightarrow} \boldsymbol{X}_S$, there are two HSCs for long and short-term sessions.

$$\boldsymbol{f}_L = HSC(\boldsymbol{h}_u, \boldsymbol{C}^0, \boldsymbol{X}_L),$$

$$\boldsymbol{f}_S = HSC(\boldsymbol{f}_L, \boldsymbol{C}^0, \boldsymbol{X}_S).$$

For the short-term session, the initial user embedding $\boldsymbol{h}_u$ is replaced by the output of the long-term HSC $\boldsymbol{f}_L$ to track more historical status of users' behavior information. Finally, the hybrid user embedding $\boldsymbol{f}_S$ is utilized to predict the next items that are most likely in $\mathcal{S}_S$.

ALGORITHM 1: KA-MemNN

```
Input: The set of examples $\mathcal{D}$, initial learning rate $\eta$, regularization parameters $\lambda_{intent}$, $\lambda_e$,
    $\lambda_\Theta$, and dimension $k$.
Output: model parameters $\Theta$ and item representations $\boldsymbol{V}$.
1: Init $\Theta$ and $\boldsymbol{V}$ from Normal Distribution $\mathcal{N}(0,1)$;.
2: repeat
3:     shuffle the set of observations $\{(u, \mathcal{S}_L, \mathcal{S}_S, i^+, i^-)\}$
4:     for each observation $(u, \mathcal{S}_L, \mathcal{S}_S, i^+, i^-)$ do
5:         compute $f_{S_u}$ according to Equation (6)
6:         compute $\hat{r}_{u,i^+}$ and $\hat{r}_{u,i^-}$ according to Equation (6)
7:         compute $\mathcal{L}_{intent\_L}$ and $\mathcal{L}_{intent\_S}$ according to Equation (3)
8:         update $\Theta$ and $\boldsymbol{V}$ by the gradient descent optimization according to Equation (8)
9:     end for
10: until convergence
11: return $\Theta$ and $\boldsymbol{V}$
```

3.7 Model Learning

After the f_S has been obtained, we further utilize it with item embeddings to predict the scores from a user to items as follows:

$$\hat{r}_{u,i} = \boldsymbol{f}_{S_u}{}^{\top}\boldsymbol{e}_i. \tag{6}$$

Next, we employ a pairwise loss function, which is introduced in [29], to train our model. We randomly select a positive example i^+ from the current session of user u, i.e., $i^+ \in \mathcal{S}_S$, and a negative example i^- from unvisited items, i.e., $i^- \in \mathcal{I}\backslash(\mathcal{S}_L \cup \mathcal{S}_S)$. Then we define a pairwise order $\hat{r}_{u,i^+} > \hat{r}_{u,i^-}$ based on a simple assumption that users prefer observed item i^+ rather than unobserved item i^-. Finally, the loss of the prediction is as follows:

$$\mathcal{L}_{pred} = \sum_{(u,\mathcal{S}_L,\mathcal{S}_S,i^+,i^-)\in\mathcal{D}} -\ln\sigma(\hat{r}_{u,i^+} - \hat{r}_{u,i^-}), \tag{7}$$

where $\mathcal{D}$ is the set of examples and it is a five-tuple denoted as (user index, a long-term session, a short-term session, a visited item index, and an unvisited item index). Combined with a constraint term of a latent intent tree, which is defined in Equation (3), the final optimization function is as follows:

$$\arg\min_{\Theta} \mathcal{L}_{pred} + \lambda_{intent}(\mathcal{L}_{intent\_L} + \mathcal{L}_{intent\_S}) + \lambda_e \parallel \boldsymbol{V} \parallel_2 + \lambda_\Theta \parallel \Theta \parallel_2, \tag{8}$$

where $\mathcal{L}_{intent\_L}$ (resp. $\mathcal{L}_{intent\_S}$) is the loss of the long (resp. short)-term session, Θ are the sets of model parameters except item embeddings $\boldsymbol{V}$, such as the parameters of MLP for users $\boldsymbol{W}_u$ and $\boldsymbol{b}_u$. The scalars λ_{intent}, λ_e, and λ_Θ are regularization parameters. The detailed learning algorithm is presented in Algorithm 1. Once the model has been learned, we recommend items with the largest scores $\hat{r}_{u,i}$ to user u.

Computational complexity: The computational complexity of KA-MemNN mainly comes from HSC, including TLWU, T, and SR parts. The complexity for both TLWU and SR is $O(batch\_size \cdot |\bar{\mathcal{S}}|\bar{n}_c k)$, where $|\bar{\mathcal{S}}|$ represents the average length of sessions, $\bar{n}_c$ represents the average number of intents in each layer, and k represents the vector dimension. In practice, $|\bar{\mathcal{S}}|$, $\bar{n}_c$, and k are typically small. As for the transformer T, the complexity is $O(|\mathcal{I}|\bar{n}_c k)$, where $|\mathcal{I}|$ represents the number of items. Thus, the overall complexity of KA-MemNN is $O(batch\_size \cdot |\bar{\mathcal{S}}|\bar{n}_c k + |\mathcal{I}|\bar{n}_c k)$.

3.8 KA-MemNN-CA: A Specific Case

The algorithm proposed in [49], which is renamed as KA-MemNN-CA in this article, also models user intents. Essentially, it is a special case of KA-MemNN. Next, we will discuss the technical relevance between KA-MemNN and KA-MemNN-CA and introduce how to generate KA-MemNN-CA according to KA-MemNN. In general, compared to KA-MemNN, KA-MemNN-CA has additional observed item-category maps and user-specific category representations. At the same time, KA-MemNN-CA removes the transformer T and intent constraint terms, i.e., $\mathcal{L}_{intent}$.

Specifically, to embody latent intents in KA-MemNN-CA, we replace intents with observed item categories since the two are highly related according to the descriptions above. Further, the map between items and categories can substitute the relations between items and latent intents. Thus, the SR can be removed from the HSC, as shown as Figure 5(b). In addition to common inputs like $\boldsymbol{h}_u$, $\boldsymbol{C}^0$, and $\boldsymbol{X}$, for an HSC with the SR removed, there are some additional inputs, i.e., $\overline{\boldsymbol{M}}$. In Figure 5(b), we should manually convert the real map between items and each layer of categories to form discrete matrices $\overline{\boldsymbol{M}}^i = [\ldots, \overline{\boldsymbol{m}}^i_j, \ldots]^\top, \forall j \in \mathcal{S}$, where $\overline{\boldsymbol{m}}^i_j$ is a multiple-hot vector like that in Equation (2). The difference is that this vector represents an observed relation instead of a latent mapping result.

Meanwhile, this is an attention module generating user-specific category representations in KA-MemNN-CA. In such a case, item embedding $\boldsymbol{e}$ is viewed as a topic probability distribution, and each category embedding $\boldsymbol{C}_{j:}$ is a summary of the items, which belong to this category. Following the attention mechanism, the importance of each item in representing their categories is automatically determined by the model as follows:

$$
\begin{aligned}
\overline{\boldsymbol{X}} &= [\ldots, \boldsymbol{e}_i, \ldots]^\top, \forall i \in \mathcal{S}_L \cup \mathcal{S}_S, \\
\overline{\boldsymbol{M}}^i &= [\ldots, \overline{\boldsymbol{m}}^i_j, \ldots]^\top, \forall j \in \mathcal{S}_L \cup \mathcal{S}_S, \\
\overline{\boldsymbol{w}}_A &= softmax(\overline{\boldsymbol{X}}\boldsymbol{h}_u), \\
\overline{\boldsymbol{C}}^i &= \overline{\boldsymbol{M}}^{i\top}(\overline{\boldsymbol{X}} \otimes \overline{\boldsymbol{w}}_A),
\end{aligned}
$$

where $\overline{\boldsymbol{X}}$ is an embedding matrix of all visited items by a user and $\overline{\boldsymbol{M}}^i$ is an item-category map matrix in the ith layer of categories. Each column of the weighted matrix, i.e., $\{\overline{\boldsymbol{X}} \otimes \overline{\boldsymbol{w}}_A\}_{:i}$, is equal to $\overline{\boldsymbol{X}}_{:i} \otimes \overline{\boldsymbol{w}}_A$.

KA-MemNN-CA can further use item categories' information based on the analogy between item categories and user intents. However, this kind of information, especially detail one, is not always available. Thus, KA-MemNN is proposed to address this most common issue by configuring the user intent as a latent contextual variable rather than a fixed item category.

4 EXPERIMENTS

In this section, we empirically verify that our proposed approach outperforms start-of-the-art baselines as well as validate that the components in our model can improve the recommendation performance regarding multiple metrics.

4.1 Experimental Setup

Datasets: Since our model fits the situation where user's purchasing and visiting behaviors are strongly associated with some inherent intentions, we choose Tmall dataset [12], Amazon dataset [10], and Gowalla dataset [4] to conduct our experiments. Tmall dataset, which is the IJCAI-15

Table 2. Statistics of Datasets

Dataset	Tmall	Gowalla	Amazon
#user	20,202	15,063	38,871
#item	24,774	11,897	61,541
#layer of categories (including item layers)	2	2	2
#category/#region	769	4,544	3
avg. session length	2.72	3.02	2.71
#(train session pair)	52,577	114,272	173,061
#(test session pair)	4,040	3,012	7,775
user-item matrix density	0.039%	0.149%	0.024%

Table 3. The Default Settings for KA-MemNN in Our Experiments

Parameter	Tmall	Gowalla	Amazon
#epoch	30	30	30
batch size	1,024	1,024	1,024
hidden size k	50	50	50
learning ratio (lr)	0.1	0.1	0.1
lr decay ratio γ	0.8	0.8	0.5
lr decay step size	30	30	20
the max length of long-term sessions	5	5	5
λ_e	0.5	0.1	0.3
λ_Θ	0.5	0.5	0.5
λ_{intent}	0.1	0.01	0.05
#layer	5	4	5

competition dataset,[1] records users' historical purchase behaviors, while Amazon dataset covers reviews from Amazon ranging from May 1996 to October 2018. Following the work [34], we mix the purchase data from multiple categories and use the review timestamps to approximate the timestamps of purchasing. Gowalla dataset[2] collects users' check-in information including the time and locations of check-ins. For Tmall and Gowalla datasets, we extract the data generated in the last seven months and items, which have been observed by more than 20 users to form the final datasets. For Amazon dataset, we remove items with fewer than 50 purchases [34]. Similar to [42], user behaviors in each day are treated as a session, while all singleton sessions, i.e., sessions containing only one item, are excluded. Besides, the user who has less than three sessions are also excluded. Any sessions except the first one of a user can be viewed as a short-term session. All sessions ahead of the short-term session can be merged as a long-term session. From each short-term session, we randomly select one item as the next item to be predicted. We treat the last sessions of the randomly selected 20% of users as short-term test sessions, and the test next predicted items are removed. Then, all short-term and correspondingly long-term sessions, i.e., session pairs, are utilized for training the model. The excluded items are utilized to examine the performance of the model. The descriptive statistics of all datasets are summarized in Table 2. Note that the categories in Tmall and Amazon and the region grids of size $1 \times 1km^2$ in Gowalla, which are divided according to the **Geographic Information System** (**GIS**) coordinates, are treated as behavioral intents in KA-MemNN-CA.

[1]https://tianchi.aliyun.com/dataset/dataDetail?dataId=47.
[2]https://snap.stanford.edu/data/loc-gowalla.html.

Settings: Models are tuned for best performance by tuning the parameters. Except for the experiments to explore parameter influences, the default settings for our model in the experiments are listed in Table 3. Specifically, we utilize Adam [20] to optimize the training process, and the initial learning rates are set to 0.1 for all datasets. We also utilize a scheduler to decay the learning rate of each parameter group by γ every step size batches.

Baselines: We compare our model with the following baseline algorithms, including non-session recommendation, session-based (sequential) recommendation, hierarchical representation approaches, and memory-based methods. A brief description of the baselines are as follows:

(1) **BPR** [29]. A traditional latent factor **collaborative filtering (CF)** model, which optimizes a pairwise loss function. We choose BPR since it is a representative baseline in many RS works. Also, we employ the pairwise loss function in our model.

(2) **CMN** [5]. A non-session memory network, which takes users' neighborhoods as the values in the memory bank. It is chosen as a baseline sine it is a state-of-the-art memory-based approach. It accumulates user preference information from the memory bank as our model. The main difference is that the memory bank in CMN has no relationship to sessions and thus it is a non-session RS. We set the number of hops to 2 in our experiments since this achieves the best performance on the datasets.

(3) **FPMC** [30]. This method models user preference by combining MF, which captures users' general preference and a first-order MC to predict users' next actions. FPMC and the following FOSSIL are two successful methods that utilize MC to model the users' sequential preference, and they are also frequently compared by other session-based RSs.

(4) **FOSSIL** [9]. This method integrates factored item similarity with MC to model a user's long- and short-term preferences. It is a state-of-the-art MC-based approach proposed in the RS area. Note that we set μ_u and μ as single scalar since the length of each session is variable.

(5) **HRM** [35]. This method generates a hierarchical user representation to capture sequential information and general tastes. The hierarchical structure of HRM is different from ours. But it also aggregates a session matrix to a vector and thus it can be viewed as a basic hierarchical model for comparison purposes. We use max pooling as the aggregation operation in our experiments because this achieves the best result.

(6) **SHAN** [42]. This is a state-of-the-art hierarchical representation method proposed in the session-based RS area. Although the structure of SHAN is not related to user intents, it consists of two elaborate attention networks that help mine users' long- and short-term preferences. Thus, SHAN can achieve competitive performance and is a strong competitor for our model.

(7) **SASRec** [19]. It adopts an enhanced self-attention neural network to capture users' long-term semantic and highlight relatively few actions in user interaction sequences simultaneously. This work is also a highly competitive baseline in sequential recommendation domains.

(8) **HyperRec** [34]. This is also a state-of-the-art sequential RS. Unlike our work, it splits user interaction sequences into multiple sequential hypergraphs and can thus model various connection information and correlations between items.

(9) **TMRN** [16]. This work is similar to ours. It is also a memory neural network and takes item categories into account. However, it combines real taxonomy information and encodes the hierarchical category semantics without incorporating item representation. Each hop is related to one layer of categories and cannot accumulate user preference information from multiple sessions. Moreover, it cannot be applied when the taxonomy information is

Table 4. Performance Comparison of Methods on Different Datasets[1]

Method	Tmall Dataset			Gowalla Dataset			Amazon Dataset		
	R@10	R@20	MRR	R@10	R@20	MRR	R@10	R@20	MRR
(a) **BPR**	0.0168	0.0196	0.0075	0.1043	0.1808	0.0737	0.1437	0.2174	0.0892
(b) **CMN**	0.0359	0.0477	0.0173	0.1400	0.2173	0.0938	0.1593	0.2487	0.1032
(c) **FPMC**	0.0860	0.1035	0.0490	0.2382	0.3147	0.1205	0.1826	0.2653	0.1361
(d) **FOSSIL**	0.0724	0.0876	0.0362	0.2052	0.2886	0.1108	0.1903	0.2713	0.1399
(e) **HRM**	0.0849	0.1074	0.0491	0.2852	0.3913	0.1660	0.2059	0.2883	0.1486
(f) **SASRec**	0.0924	0.1248	0.0523	0.2998	0.4004	0.1736	0.2246	0.3097	0.1610
(g) **SHAN**	0.1087	0.1425	0.0532	0.3119	0.4158	0.1804	0.2361	0.3188	0.1732
(h) **HyperRec**	0.1003	0.1436	0.0528	0.3294	0.4203	0.1895	0.2510	0.3297	0.1843
(i) **TMRN**	<u>0.1098</u>	<u>0.1437</u>	<u>0.0539</u>	<u>0.3321</u>	<u>0.4365</u>	<u>0.1987</u>	<u>0.2598</u>	<u>0.3356</u>	<u>0.1942</u>
(j) **KA-MemNN-CA**	0.1201	0.1554	0.0581	**0.3687**	0.4838	**0.2268**	0.2691	0.3405	0.2097
(k) **KA-MemNN**	**0.1334***	**0.1619***	**0.0612**†	0.3676*	**0.4857***	0.2245*	**0.2794***	**0.3587***	**0.2201***
Improvement: (k) vs. (i)	21.49%	12.67%	13.54%	10.69%	11.27%	12.98%	7.54%	6.88%	13.34%

[1]The best result in each column is boldfaced, and the best result except the proposed methods in each column is underlined. * indicates that the improvement of KA-MemNN is statistically significant compared with the underlined result with a corrected $p < 0.05/9$; † indicates the improvement is statistically significant without corrections (i.e., $p < 0.05$).

unknown. In this article, we set the number of hops to 1 as we do not have detailed taxonomy information.

Metrics: We employ two commonly used metrics, Recall@K (i.e., R@K for short in Table 4) and MRR , to evaluate the performance of the methods. Recall@K evaluates the fraction of ground truth items that have been retrieved in a top@K item list over the total amount of ground truth items. We choose $K \in \{10, 20\}$ because most users are only interested in viewing the recommendation on the first page in real-world RSs [12]. In our case, since the number of ground truth items for each user is 1, i.e., one next item to be predicted, the value of Recall is equal to the value of hit ratio. For top-N RSs, the position-based metrics, e.g., MRR, are more practical [48]. Thus, we also choose MRR to evaluate the position of the next item that should be recommended. The larger the values of both Recall and MRR metrics, the better the performance.

4.2 Comparison of Performance

Table 4 compares the performance of our KA-MemNN model and the baselines on three datasets regarding two metrics. From the table, we make the following key observations:

(1) Our proposed methods KA-MemNN and KA-MemNN-CA outperform all baselines including non-session approaches, i.e., BPR and CMN, traditional MC-based next item recommendation methods, i.e., FPMC and FOSSIL, the hierarchical representation methods, i.e., HRM and SHAN, the state-of-the-art sequential methods, i.e., SASRec and HyperRec, and a taxonomy-aware MemNN method, i.e., TMRN, on all datasets. For example, at Recall@10, KA-MemNN achieves an improvement of 21.49%, 10.69%, and 7.54% compared with the second-best baseline, i.e., TMRN, on the Tmall, Gowalla, and Amazon datasets, respectively, although TMRN also achieves excellent performance. For the MRR metric, KA-MemNN also improves the performance by 13.54%, 12.98%, and 13.34% on the three datasets, respectively. This observation empirically verifies the superiority of our proposed KA-MemNN concerning the next recommendation problems with respect to Recall and MRR metrics. We also employ the Wilcoxon matched-pairs signed-rank test to evaluate the difference between the performance of TMRN (i.e., the best one except the proposed methods) and KA-MemNN. The p-values of all three metrics (i.e., R@10, R@20, and MRR) on three datasets are less than 0.05. After correcting for multiple testing, although some improvements cannot meet the

severe condition, i.e., $p < 0.05/9$, shown as Table 4, it cannot much affect the conclusion that the performance promotion of our model is statistically significant on multiple aspects.

(2) KA-MemNN-CA and TMRN utilize real-world taxonomy information and achieve a better performance than the other baselines. This improvement might lie in the fact that the auxiliary data can enrich the representations of users. Even so, TMRN cannot fully exploit the link information from user-category-relations as KA-MemNN-CA does, resulting in KA-MemNN-CA outperforming TMRN. The way to utilize item categories in KA-MemNN-CA and TMRN is different. Thus, an interesting work is further incorporating the information from item categories to KA-MemNN-CA following the strategy of TMRN.

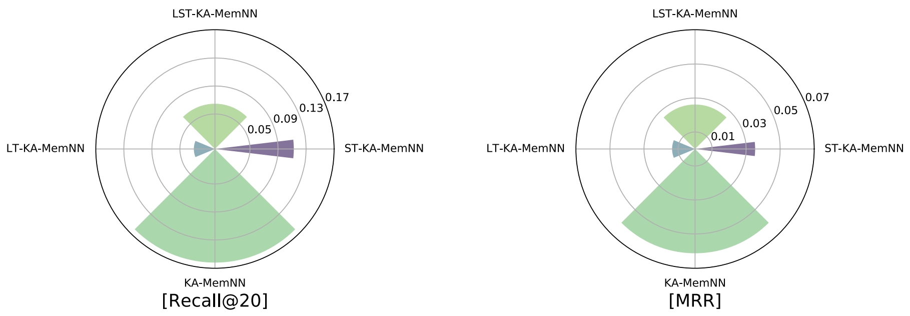
(3) In some cases, e.g., for all metrics on the Tmall dataset, KA-MemNN can achieve a better performance than KA-MemNN-CA, though the latter utilizes more additional side information than the former. This proves that our latent intent mechanism works well in KA-MemNN. Basically, KA-MemNN has a more complex network structure than KA-MemNN-CA since the latter is a specific case of the former. KA-MemNN can fill in the gap caused by the lack of real-world taxonomy information through an elaborate design of the model. This is one possible reason why KA-MemNN performs better than KA-MemNN-CA and TMRN. The other reason might be that we only have one layer of real item categories, which might affect KA-MemNN-CA and TMRN's performance. But in reality, the taxonomy information, especially detailed multi-level category information, is not always available. KA-MemNN is qualified for this non-ideal situation and even works better in some cases by introducing a latent intent tree. However, if the auxiliary information is accumulated and collected sufficiently, we will further explore the performance of KA-MemNN-CA and the performance difference between KA-MemNN and KA-MemNN-CA. This work can help identify the essential relationship between item categories and latent user intents.

(4) The performance of session-based methods including KA-MemNN, KA-MemNN-CA, TMRN, SHAN, HRM, SASRec, and HyperRec, is better than those of the BPR and CMN methods, which neglect the sequential information, depicted by a substantial improvement gap, e.g., 14.23% on Tmall, 30.49% on Gowalla, and 14.13% on Amazon for Recall@20 between KA-MemNN and BPR. This improvement indicates that the combination of users' long- and short-term behaviors is necessary to promote recommendation performance. The users' current intents and preferences have a significant influence on the users' next actions. In a statistical sense, the probability distribution of user behaviors during different periods is volatile. Therefore, the indiscriminate utilization of old and new samples might have negative effects on the performance of models. However, conventional non-session RSs like BPR and CMN mix a user's all interacted items into one box without discrimination and discards time information. This setting impairs the performance of recommendations. Fortunately, the long- and short-term session mechanism can relieve this problem.

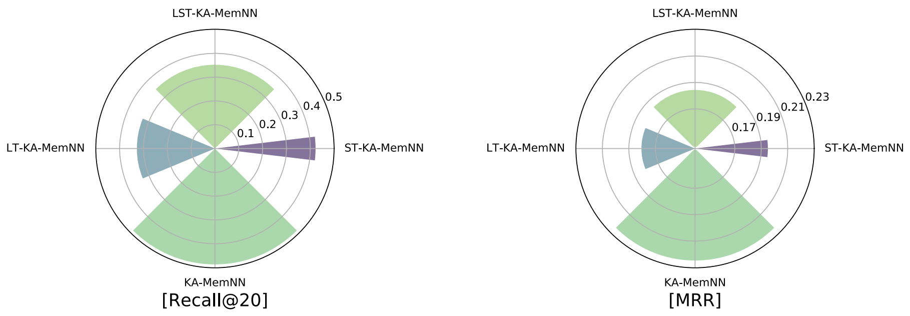
(5) The performance of CMN is better than that of BPR, although both are non-session approaches. The reason for this may lie in the fact that CMN accumulates useful knowledge from the memory banks to form informative representations of users, which verifies the effectiveness of MemNNs as well. The memory mechanism in CMN can help incorporate more local information by retrieving the nearest or the most relevant entries. By contrast, BPR only considers the collaborative filtering relations between users and items from a global perspective. Thus, BPR would be biased to dominant entries (e.g., active users and popular items). The problem of data sparsity and data imbalance will cause the performance of these RSs to decrease significantly.

4.3 Ablation Study

In these parts, we conduct a series of ablation tests by removing or restraining the contributions of main components in our model to evaluate their influences. Since Tmall and Amazon have similar



(a) Tmall dataset



(b) Gowalla dataset

Fig. 6. Experimental results for exploring the impact of long- and short-term sessions. The value of the radius of a circular sector indicates the performance of the metrics and the size of the central angle indicates the percentage of users' behavioral data that has been used.

experiment observations and to save space, in the following subsections, we only show the results on Tmall and Gowalla datasets.

4.3.1 Influence of Long- and Short-Term Components. To explore the influence of long- and short-term KA-MBs, we further propose the additional three simplified versions of KA-MemNN, namely LT-KA-MemNN, ST-KA-MemNN, and LST-KA-MemNN. Compared to the framework of KA-MemNN, the simplified versions only have one hop, e.g., 1st or 2nd HOP in Figure 2. More specifically, the framework of LT-KA-MemNN and ST-KA-MemNN only has long-term sessions or short-term sessions, respectively. For LST-KA-MemNN, we merge the long- and short-session of a user to one session, i.e., $\mathcal{S}_{LS} = \mathcal{S}_L \cup \mathcal{S}_S$ and correspondingly $\mathcal{S}_{LS} \xrightarrow{generate} X_{LS}$.

Figure 6 shows the experimental results of the three simplified approaches along with the complete version of KA-MemNN. First, we list two key properties according to the experimental settings: (1) ST-KA-MemNN utilizes more newly generated data to train the model compared with LT-KA-MemNN. At the same time, the volume of the training set for LT-KA-MemNN is larger than that of ST-KA-MemNN. (2) LST-KA-MemNN and KA-MemNN have the same training set, including the parts utilized by LT-KA-MemNN and ST-KA-MemNN. Then, based on Figure 6, we draw the following possible conclusions:

(1) The approaches fed with more or newer users' behavior data generally perform better. For example, the performance of ST-KA-MemNN and LST-KA-MemNN is better than that of LT-KA-MemNN on both datasets in terms of Recall@20 and MRR metrics.

Table 5. Results for Ablation Study

Architecture	Tmall	Tmall	Gowalla	Gowalla
	Recall@20	MRR	Recall@20	MRR
(a) KA-MemNN	0.1619	0.0612	0.4857	0.2245
(b) (-) Intent	0.1500	0.0511	0.4657	0.2099
(c) (-) Equation (3)	0.1516	0.0561	0.4687	0.2131
(d) KA-MemNN-Avg	0.1612	0.0601	0.4726	0.2149

(-) Denotes Removing the Specific Component.

(2) Treating the long- and short-term sessions differently can promote the performance of models. For instance, KA-MemNN performs better than LST-KA-MemNN on both datasets, though both utilize long- and short-term sessions. The only different but critical design is that KA-MemNN employs an independent hop to model long-term sessions. The following observations further verify this conclusion.

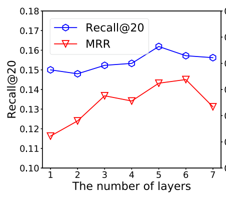
(3) Concept drift might exist in both the Tmall and Gowalla datasets because the performance of ST-KA-MemNN is better than LST-KA-MemNN. Although the training set of LST-KA-MemNN includes that of ST-KA-MemNN, the older examples might suppress the model to learn the users' current interests. This is why we introduce a long- and short-term mechanism to session-based RSs.

(4) The preferences of users on Tmall are more sensitive to time changes than those of the users on Gowalla. For instance, the performance gap between LT-KA-MemNN and LST-KA-MemNN on the Tmall dataset is much larger than that on the Gowalla dataset. For Recall@20 and MRR metrics, the short-term sessions can improve the performance by 114% and 99% on the Tmall dataset, respectively. In comparison, the improvements are, respectively, 7% and 2% for Recall@20 and MRR metrics on the Gowalla dataset.

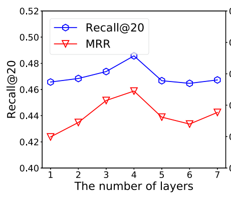
4.3.2 Influence of Other Main Components. Table 5 further shows the results for additional ablation study. We observe that our KA-MemNN model outperforms any other variants, indicating the effectiveness of the designed architecture. In the (b) line, we remove the intent layer in our model. That is, there is only the first layer in TLWU. The output of TLWU in Equation (5) is redefined as $\boldsymbol{o} = \boldsymbol{X}^{\top}\boldsymbol{w}_A$. Then, the HSC and TLWU become a normal attention mechanism that aggregates a session matrix $\boldsymbol{X}$ to a vector by a weighted combination of all item representations in that session. However, this setting decreases the value of Recall@20 and MRR metrics. In the (c) line, we set $\lambda_{intent} = 0$ to remove the constraint term defined in Equation (3), leading to the model has no component shaping intent representations or enhancing the relationship between different layers. However, this setting also impairs the model's performance, thus it is necessary to add this constraint term with a tuned λ_{intent} to the final optimization task. In the (d) line, we replace the weighted pooling in the first layer of TLWU with an average pooling. The average pooling assumes all items have the same weight, thus, it is a specific case of the weighted pooling. This setting limits the capabilities of the model but affects the performance slightly. There are other aggregation functions such as max pooling, Hadamard product, and set transformer [22]. They might have different motivations behind the aggregation mechanism from our TLWU, thus we do not study these operations in this work.

4.4 Analysis of Parameter Influence

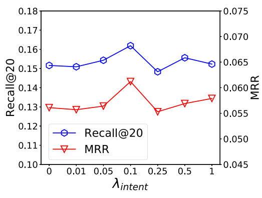
In this subsection, we explore the performance of our model with different values of various parameters and the results are shown in Figure 7.



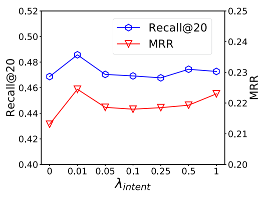
(a) Tmall: layers



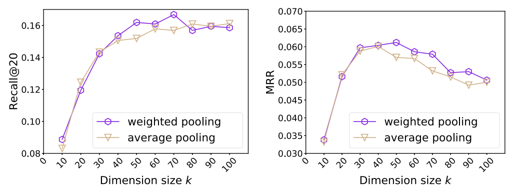
(b) Gowalla: layers



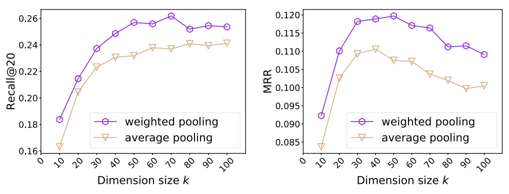
(c) Tmall: λ_{intent}



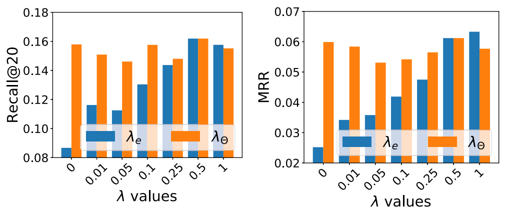
(d) Gowalla: λ_{intent}



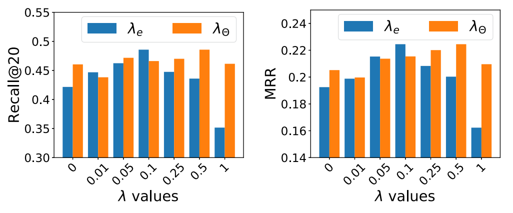
(e) Tmall: poolings and dimensions



(f) Gowalla: poolings and dimensions,



(g) Tmall: λ_e and λ_Θ



(h) Gowalla: λ_e and λ_Θ

Fig. 7. Results of experiments exploring the impact of various parameters.

4.4.1 Influence of the Number of Latent Intent Layers. To gain further insight into the influence of the model structure, this subsection explores the performance of KA-MemNN across different numbers of latent intent layers and shows the experimental results in Figure 7(a) and Figure 7(b). In real-world applications, the number of category layers is limited, thus we set the value of the number of intent layers from 1 to 7 for both the Tmall and Gowalla datasets. When the value is set to 1, the HSC and TLWU are degraded to a normal attention mechanism as aforementioned.

From Figure 7(a), we can observe that: on the Tmall dataset, almost all the settings where the number of layers is larger than one improves the performance compared to the setting with the one-layer structure, i.e., the number of layers is 1. For example, the best performance, i.e., where the number of layers is 6, can increase the value of MRR by 21.14%. For the Recall@20 metric, the best point is 5, which is the default setting in KA-MemNN, and it improves the performance by 7.93%. On the Gowalla dataset, which is shown as Figure 7(b), the best choice for the number of layers is 4 for both the Recall@20 and MRR metrics. Similarly, the model with the multiple latent intent layers works better than the model with only the one layer of item representations. These observations prove the effectiveness of the hierarchical modeling of user preference with coarse-to-fine grained intents considered.

4.4.2 Influence of the Parameter λ_{intent}. The subfigures Figure 7(c) and Figure 7(d) show the experimental results across different values of the parameter λ_{intent}, which is described in Equation (8), on the Tmall and Gowalla datasets, respectively. The value of λ_{intent} decides how important the constraint term related to the latent intent tree is to the model. On the Tmall dataset, when $\lambda_{intent} = 0.1$, the model achieves the best performance, whereas on the Gowalla dataset, the best choice for λ_{intent} is 0.01. Additionally, when $\lambda_{intent} = 0$, there is no constraint term, causing performance degradation as aforementioned.

4.4.3 Influence of Pooling Functions and Dimension Sizes. This part compares the performance of the original KA-MemNN and its variant employing an average pooling in the first layer of TLWU. Figure 7(e) and Figure 7(f) show the values of Recall@20 and MRR for KA-MemNN across a different number of dimensions with size k from 10 to 100, and also show the performance of KA-MemNN when it employs different aggregation operations. Accordingly, we make four major observations, as follows:

(1) A larger value of k can improve the performance since it increases the representation capability of the model.

(2) When the dimension size k is small, e.g., $k \leq 30$, the model doesn't reach the best performance, thus the difference in the performance caused by the pooling function selection is not remarkable. Dimension size k is the key factor leading to the performance difference during this stage.

(3) When dimension size k is large enough, e.g., $k \geq 50$, the influence from the dimension size becomes limited. Meanwhile, a performance gap between weighted pooling and average pooling settings has emerged.

(4) Weighted pooling achieves a better performance than average pooling when the impact of the dimension size k is largely eliminated. This indicates that the architecture of TLWU is effective. Weighted pooling causes our model to weight the items twice; the lower layer's weights are the linking strength between the items and the target user, and the upper layer's weights are the linking strength between the intents and the target user. This weighting mechanism depicts user-intent-item relations better and exploits the full modeling capacity of TLWU.

4.4.4 Influence of Parameters λ_e and λ_Θ. Parameters λ_e and λ_Θ, which are similar to λ_{intent}, are regularization hyperparameters. To simplify the analysis, we assume the impact of λ_e and λ_Θ is independent. In each experiment setting, we fix one of them to the best/default value, and then explore the impact of the other one. The experimental results are shown in Figure 7(g) and Figure 7(h). We observe that the impact of λ_Θ seems like no significant pattern. But it still greatly affects the performance of the model. For instance, the biggest performance gap caused by λ_Θ is 2.48% for MRR on the Gowalla dataset. The best performance is improved by 12.42% compared with the worst performance.

In contrast, the impact of λ_e has an optimal point (turning point), e.g., $\lambda_e = 0.5$ for Recall@20 on the Tmall dataset and $\lambda_e = 0.1$ for Recall@20 on the Gowalla dataset. The settings that are larger or smaller than the optimal point deteriorate the performance of our model. Furthermore, according to our observations from the subfigures, the model might be easier to over-fit the training set of the Tmall dataset compared to that of the Gowalla dataset. Because the setting $\lambda_e = 0$, i.e., there is no regularization term on item representations $\boldsymbol{V}$, significantly decreases the values of the two metrics on the Tmall dataset while the impact on the Gowalla dataset is relatively limited. Furthermore, the optimal point of λ_e on the Tmall dataset, i.e., $\lambda_e = 0.5$, is larger than that on the Gowalla dataset, i.e., $\lambda_e = 0.1$. These observations further validate that concept drift is more serious in the Tmall dataset than the Gowalla dataset. Therefore, compared to the Tmall dataset, the Gowalla dataset has a more identical distribution between the training set, i.e., users' older behaviors, and the test set, users' current behaviors. It is also the reason why the values of Recall and MRR metrics on the Gowalla dataset are much larger than those on the Tmall dataset.

4.4.5 Influence of the Maximum Length of Long-Term Sessions. Since we merge all sessions before the short-term session to become the long-term session, some users' long-term sessions might contain a huge volume of data. To overcome the unbalanced data problem and speed up the training process, we introduce a parameter named **Max LT Length** to control the maximum length of the long-term sessions. Specifically, for the long-term sessions with a large volume of data, we

Table 6. Performance of KA-MemNN on Different Maximum Lengths of Long-Term Sessions

Max LT Length	**Tmall Dataset**			**Gowalla Dataset**		
	Recall@10	Recall@20	MRR	Recall@10	Recall@20	MRR
5	**0.1334**	**0.1619**	**0.0612**	**0.3676**	**0.4857**	**0.2245**
10	0.1298	0.1500	0.0576	0.3585	0.4710	0.2125
15	0.1311	0.1602	0.0596	0.3406	0.4544	0.2136

The best result in each column is boldfaced.

randomly sample **Max LT Length** items to re-form them. Note that the **Max LT Length** only works in the training process. In the testing stage, we still keep all the visited items to provide as much information as possible.

Table 6 shows the experimental results across different maximum lengths of long-term sessions. As we can see from the table, on both the Tmall and Gowalla datasets, a smaller value of **Max LT Length** instead of a larger one results in the best model performance. A possible reason for this is that the average lengths of the sessions on the Tmall and Gowalla datasets are 2.72 and 3.02, respectively. Most users only have 2 to 3 sessions. Therefore, limiting the length of long-term sessions to five can help the model reduce or avoid bias toward very active users. Moreover, the impact of different values of **Max LT Length** is limited. Thus, we choose five for both datasets to speed up the training process.

4.5 Visualization of Attention

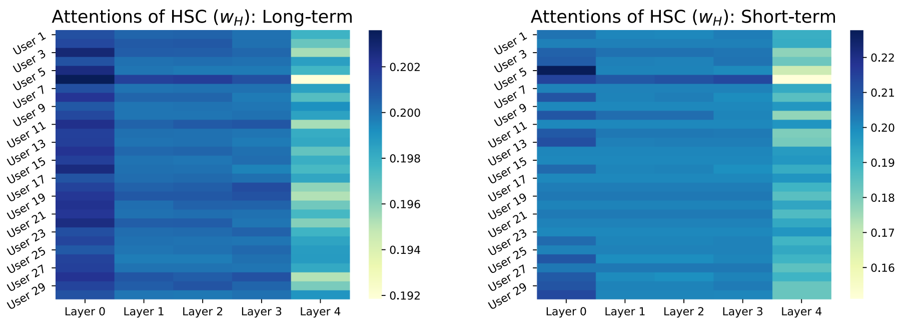
The visualization of attention can help to analyze how the model works. Therefore, this subsection plots multiple heat maps to visualize some critical attentions/weights produced in KA-MemNN. The trained model is fed with the test set, and then all the heat maps are plotted based on the intermediate records of the model.

The attentions of HSC. We randomly select 30 users from both the Tmall and Gowalla datasets. Then, for each user, we visualize the weight distributions on the layers of HSC. The heat maps are shown in Figure 8 in which the figures of the long- and short-term sessions are quite similar. Moreover, there are another two main observations, which are as follows:

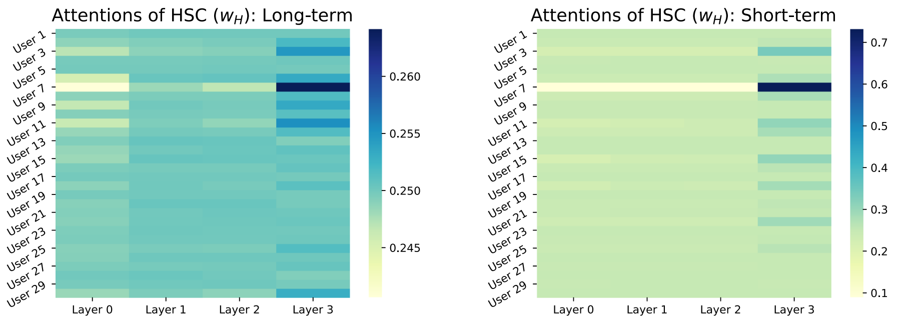
(1) The weight distribution varies between users. Some users, such as user 5 of the Tmall dataset, pay more attention to layer 0, which represents specific items, while some other users, such as user 7 of the Gowalla dataset, mainly focus on a high-level intent, e.g., layer 3. This is understandable in terms of reason and common sense as users might have different decision-making processes. HSC can automatically decide which level intent is more important to a user.

(2) Generally, for users of the Tmall dataset, the most fine-grained layers, e.g., Layer 0, have a relatively large impact on the combined outputs of HSC. On the contrary, for users of the Gowalla dataset, the most coarse-grained layers, e.g., Layer 3, are more important. Maybe there is a strong relationship between intents and items but a weak relationship between items on the Gowalla dataset, whereas on the Tmall dataset, the situation is the opposite.

The attentions of TLWU. This part shows the visualization of user-intent distributions $\boldsymbol{w}_K$, soft maps between items and intents $\boldsymbol{M}$, user-item distributions $\boldsymbol{w}_A$, and adjusted user-item distributions. We take layer 3 and layer 2 of HSCs for the Tmall and Gowalla datasets as an example, respectively. Then, the number of intents in the corresponding layers is 12 and 22, shown in the left two subfigures in Figure 9. We can see from these two subfigures that users have different intent distributions. The non-zero intents are indexed according to the visited items and the map between items and intents shown as the center two subfigures in Figure 9. For instance, user 4 in Figure 9(b) has visited items 1 and 2, which cover the intents in the set $\{6, 7, 8, 10, 11, 12, 15, 16, 17, 21, 22\} = \{6, 7, 8, 10, 11, 12, 15, 17, 21, 22|\text{item1}\} \cup \{7, 16, 17|\text{item2}\}$, shown as the center subfigure. Then, user

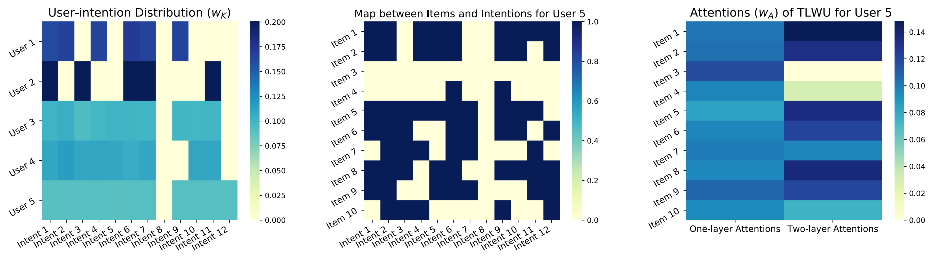


(a) Tmall dataset

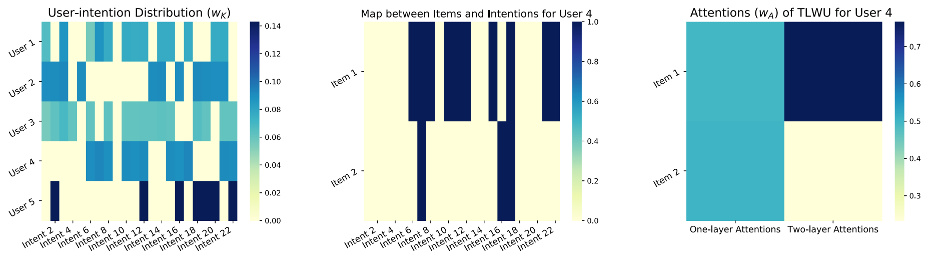


(b) Gowalla dataset

Fig. 8. The heat map of attentions of each layer in HSC. The color scale indicates the value of the weights, darker representing a higher weight and lighter a lower weight.



(a) Tmall dataset: Layer 3 of HSC



(b) Gowalla dataset: Layer 2 of HSC

Fig. 9. The visualization of $\mathbf{w}_K$, $\mathbf{w}_A$, and $\boldsymbol{M}$ in TLWU for short-term sessions.

4's line in the left subfigure exhibits the weights of all the involved intents. Without considering the influence of intent, we can directly calculate the attentions of each item in a session, i.e., $\boldsymbol{w}_A$ shown in the first column of the right subfigure. From the subfigure, we observe that for user 4, items 1 and 2 are of nearly equal importance. According to our design, each item's importance is re-adjusted by considering the user-intent distribution. The adjusted user-item distributions are shown in the second column of the right subfigure. From the subfigure, we find that the importance of item 1 is amplified since it conforms to more user intents. A similar observation is also shown in Figure 9(a). For example, the importance of item 3 to user 5 decreases since the item does not match any intents of the user.

The following gives a more detailed explanation of how TLWU works in mathematical form. Assume each item has an original weight α_i, i.e., the element in $\boldsymbol{w}_A$, and the item matches multiple user intents with different weights β_j, i.e., the element in $\boldsymbol{w}_K$. Then, the re-adjusted weight of each item is $\alpha_i \sum_j \beta_j$. Thus, according to this design, the more user intents the item matches, and the higher the weight of the matching intent, the more valuable the item is in representing the session.

Moreover, through analyzing the attention for customized category representations in KA-MemNN-CA, we find that the frequently visited items usually obtain a relatively larger weight when generating their category embeddings. Also, the items, which have been visited in a recent session also have a larger impact on their category embeddings compared to previously visited items. This phenomenon may be reasonable since category-specific users' preferences are reflected in the frequently visited items that belong to this category, and users' current intents usually have a strong relation with recently visited items.

5 CONCLUSION

In this article, we viewed a user's session as a set and left each item's weight to be learned automatically. Different from existing works, we enhanced traditional weighting mechanisms by creatively introducing an intent structure above items. The user intent distribution is utilized to adjust items' original weights. This two-layer weighting mechanism is precisely in line with the user's decision-making process and increases the modeling capability.

Based on the analogy between item categories and user intents, we designed several dedicated components to shape multi-level intents' structure and build the connection between intents and items. First, there is a structure-related constraint shaping the intent tree. Second, TLWU is responsible for conducting a double weighting operation by hierarchically modeling user intents and preferences. Third, HSC decides the importance of different layers of intents. Last, by applying an HSC to long- and short-term sessions, respectively, our model can retrieve and accumulate related information from the KA-MB twice to form the final hybrid user embeddings. With a few modifications, our KA-MemNN can be applied to a specific case where item categories embody user intents, and the modified version is named KA-MemNN-CA.

Comparison experiments on real-world datasets validate that our models outperform the state-of-the-art approaches with a significant margin in terms of Recall and MRR metrics. The comprehensive ablation study, parameter analysis, and visualization prove that the design of the components such as HSC and TLWU in our model are effective and work as claimed.

Learning hierarchical intent models provides more chances to explore novel recommendation approaches. Firstly, we can integrate the intent model to existing typical RSs to further fulfill the potential of conventional methods. Secondly, we can explore more intent learning and modeling techniques. An interesting topic is that we can make use of users' search queries and discriminative behaviors to construct more informative user intents [17]. In KA-MemNN, we only explicitly recorded and learned the map between items and latent intents in different layers. However, a challenging task is how to explicitly highlight and shape the parent-child relationship between user intents in different layers. In this article, an intent model has a tree structure but it is also possible to apply other structures to model intents. Moreover, it can be useful when some mathematical properties of user intents can be defined and applied in intent modeling. Thirdly, in this article, we treated the intent as a contextual variable; however, the intent can also be viewed as a predictive factor. To predict a user's intent would be valuable for practical recommendation applications.

REFERENCES

[1] Buru Chang, Yonggyu Park, Donghyeon Park, Seongsoon Kim, and Jaewoo Kang. 2018. Content-aware hierarchical point-of-interest embedding model for successive POI recommendation. In *Proceedings of the International Joint Conference on Artificial Intelligence*. AAAI Press, 3301–3307. DOI: https://doi.org/10.24963/ijcai.2018/458

[2] Liang Chen, Yang Liu, Xiangnan He, Lianli Gao, and Zibin Zheng. 2019. Matching user with item set: Collaborative bundle recommendation with deep attention network. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2095–2101. DOI: https://doi.org/10.24963/ijcai.2019/290

[3] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the ACM International Conference on Web Search and Data Mining*. ACM, 108–116. DOI: https://doi.org/10.1145/3159652.3159668

[4] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. 2011. Friendship and mobility: User movement in location-based social networks. In *Proceedings of the SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1082–1090. DOI: https://doi.org/10.1145/2020408.2020579

[5] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative memory network for recommendation systems. In *Proceedings of the International SIGIR Conference on Research & Development in Information Retrieval*. ACM, 515–524. DOI: https://doi.org/10.1145/3209978.3209991

[6] Lei Guo, Hongzhi Yin, Qinyong Wang, Tong Chen, Alexander Zhou, and Nguyen Quoc Viet Hung. 2019. Streaming session-based recommendation. In *Proceedings of the International SIGIR Conference on Research & Development in Information Retrieval*. ACM, 1569–1577. DOI: https://doi.org/10.1145/3292500.3330839

[7] Simon Haykin and Neural Network. 2004. A comprehensive foundation. *Neural Networks* 2, 2004 (2004), 41.

[8] Ruining He, Chunbin Lin, Jianguo Wang, and Julian J. McAuley. 2016. Sherlock: Sparse hierarchical embeddings for visually-aware one-class collaborative filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence*. IJCAI/AAAI Press, 3740–3746. DOI:http://www.ijcai.org/Abstract/16/526.

[9] Ruining He and Julian McAuley. 2016. Fusing similarity models with Markov chains for sparse sequential recommendation. In *Proceedings of the International Conference on Data Mining*. IEEE Computer Society, 191–200. DOI: https://doi.org/10.1109/ICDM.2016.0030

[10] Ruining He and Julian J. McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the International Conference on World Wide Web*. ACM, 507–517. DOI: https://doi.org/10.1145/2872427.2883037

[11] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *Proceedings of the International Conference on Learning Representations*. Retrieved from http://arxiv.org/abs/1511.06939.

[12] Liang Hu, Longbing Cao, Shoujin Wang, Guandong Xu, Jian Cao, and Zhiping Gu. 2017. Diversifying personalized recommendation with user-session context. In *Proceedings of the International Joint Conference on Artificial Intelligence*. AAAI Press, 1858–1864. DOI: https://doi.org/10.24963/ijcai.2017/258

[13] Liang Hu and Soingkui Jian. 2018. Interpretable recommendation via attraction modeling: Learning multilevel attractiveness over multimodal movie contents. In *Proceedings of the International Joint Conference on Artificial Intelligence*. AAAI Press, 3400–3406. DOI: https://doi.org/10.24963/ijcai.2018/472

[14] Liang Hu, Songlei Jian, Longbing Cao, and Qingkui Chen. 2018. Interpretable recommendation via attraction modeling: Learning multilevel attractiveness over multimodal movie contents. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 3400–3406. DOI: https://doi.org/10.24963/ijcai.2018/472

[15] Liang Hu, Songlei Jian, Longbing Cao, Q Chen, and Z. Gu. 2019. HERS: Modeling influential contexts with heterogeneous relations for sparse and cold-start recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press, 3830–3837. DOI: https://doi.org/10.1609/aaai.v33i01.33013830

[16] Jin Huang, Zhaochun Ren, Wayne Xin Zhao, Gaole He, Ji-Rong Wen, and Daxiang Dong. 2019. Taxonomy-aware multi-hop reasoning networks for sequential recommendation. In *Proceedings of the ACM International Conference on Web Search and Data Mining*. ACM, 573–581. DOI: https://doi.org/10.1145/3289600.3290972

[17] Jizhou Huang, Wei Zhang, Yaming Sun, Haifeng Wang, and Ting Liu. 2018. Improving entity recommendation with search log and multi-task learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*. AAAI Press, 4107–4114. DOI: https://doi.org/10.24963/ijcai.2018/571

[18] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving sequential recommendation with knowledge-enhanced memory networks. In *Proceedings of the International SIGIR Conference on Research & Development in Information Retrieval.* ACM, 505–514. DOI: https://doi.org/10.1145/3209978.3210017

[19] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive sequential recommendation. In *Proceedings of the International Conference on Data Mining*. IEEE Computer Society, 197–206. DOI: https://doi.org/10.1109/ICDM.2018.00035

[20] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations.* Retrieved from http://arxiv.org/abs/1412.6980.

[21] Dejiang Kong and Fei Wu. 2018. HST-LSTM: A hierarchical spatial-temporal long-short term memory network for location prediction. In *Proceedings of the International Joint Conference on Artificial Intelligence.* AAAI Press, 2341–2347. DOI: https://doi.org/10.24963/ijcai.2018/324

[22] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. 2018. Set transformer: a framework for attention-based permutation-invariant neural networks. In *Proceedings of the International Conference on Machine Learning (PMLR'19)*, Vol. 97. 3744–3753.

[23] Xin Li, Dongcheng Han, Jing He, Lejian Liao, and Mingzhong Wang. 2019. Next and next new POI recommendation via latent behavior pattern inference. *ACM Transactions on Information Systems* 37, 4 (2019), 46:1–46:28. DOI: https://doi.org/10.1145/3354187

[24] Yuanzhi Li and Yang Yuan. 2017. Convergence analysis of two-layer neural networks with relu activation. In *Proceedings of the Advances in Neural Information Processing Systems*, 597–607. Retrieved from http://papers.nips.cc/paper/6662-convergence-analysis-of-two-layer-neural-networks-with-relu-activation.

[25] Zhi Li, Hongke Zhao, Qi Liu, Zhenya Huang, Tao Mei, and Enhong Chen. 2018. Learning from history and present: Next-item recommendation via discriminatively exploiting user behaviors. In *Proceedings of the SIGKDD International Conference on Knowledge Discovery & Data Mining.* ACM, 1734–1743. DOI: https://doi.org/10.1145/3219819.3220014

[26] Chen Lin, Xiaolin Shen, Si Chen, Muhua Zhu, and Yanghua Xiao. 2019. Non-Compensatory psychological models for recommender systems. In *Proceedings of the AAAI Conference on Artificial Intelligence.* AAAI Press, 4304–4311. DOI: https://doi.org/10.1609/aaai.v33i01.33014304

[27] Chen Ma, Peng Kang, and Xue Liu. 2019. Hierarchical gating networks for sequential recommendation. In *Proceedings of the SIGKDD International Conference on Knowledge Discovery & Data Mining.* ACM, 825–833. DOI: https://doi.org/10.1145/3292500.3330984

[28] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* The Association for Computational Linguistics, 1400–1409. DOI: https://doi.org/10.18653/v1/d16-1147

[29] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence.* AUAI Press, 452–461.

[30] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the WWW Conference on the World Wide Web.* ACM, 811–820. DOI: https://doi.org/10.1145/1772690.1772773

[31] Shuo Shang, Ruogu Ding, Kai Zheng, Christian S. Jensen, Panos Kalnis, and Xiaofang Zhou. 2014. Personalized trajectory matching in spatial networks. *The VLDB Journal–The International Journal on Very Large Data Bases* 23, 3 (2014), 449–468.

[32] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, Arthur D. Szlam. 2015. End-to-end memory networks. In *Proceedings of the 28th International Conference on Advances in Neural Information Processing Systems.* 2440–2448. Retrieved from http://papers.nips.cc/paper/5846-end-to-end-memory-networks.

[33] Zhu Sun, Guibing Guo, and Jie Zhang. 2017. Learning hierarchical category influence on both users and items for effective recommendation. In *Proceedings of the Symposium on Applied Computing*. ACM, 1679–1684. DOI: https://doi.org/10.1145/3019612.3019763

[34] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-item recommendation with sequential hypergraphs. In *Proceedings of the International Conference on Research and Development in Information Retrieval, SIGIR.* ACM, 1101–1110. DOI: https://doi.org/10.1145/3397271.3401133

[35] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the International SIGIR Conference on Research & Development in Information Retrieval.* ACM, 403–412. DOI: https://doi.org/10.1145/2766462.2767694

[36] Shoujin Wang, Liang Hu, Yan Wang, Quan Z. Sheng, Mehmet A. Orgun, and Longbing Cao. 2019. Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks. In *Proceedings of the International Joint Conference on Artificial Intelligence.* 3771–3777. DOI: https://doi.org/10.24963/ijcai.2019/523

[37] Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. 2018. TEM: Tree-enhanced embedding model for explainable recommendation. In *Proceedings of the WWW Conference on World Wide Web*. ACM, 1543–1552. DOI: https://doi.org/10.1145/3178876.3186066

[38] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent recommender networks. In *Proceedings of the International Conference on Web Search and Data Mining*. ACM, 495–503. DOI: https://doi.org/10.1145/3018661.3018689

[39] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Jiajie Xu, Victor S. Sheng S. Sheng, Zhiming Cui, Xiaofang Zhou, and Hui Xiong. 2019. Recurrent convolutional neural network for sequential recommendation. In *Proceedings of the WWW Conference on World Wide Web*. ACM, 3398–3404. DOI: https://doi.org/10.1145/3308558.3313408

[40] Ghim-Eng Yap, Xiao-Li Li, and S. Yu Philip. 2012. Effective next-items recommendation via personalized sequential pattern mining. In *Proceedings of the International Conference on Database Systems for Advanced Applications*. Springer, 48–64. DOI: https://doi.org/10.1007/978-3-642-29035-0_4

[41] Haochao Ying, Jian Wu, Guandong Xu, Yanchi Liu, Tingting Liang, Xiao Zhang, and Hui Xiong. 2019. Time-aware metric embedding with asymmetric projection for successive POI recommendation. *World Wide Web* 22, 5 (2019), 2209–2224. DOI: https://doi.org/10.1007/s11280-018-0596-8

[42] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential recommender system based on hierarchical attention networks. In *Proceedings of the International Joint Conference on Artificial Intelligence*. AAAI Press, 3926–3932. DOI: https://doi.org/10.24963/ijcai.2018/546

[43] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. 2018. Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine* 13, 3 (2018), 55–75.

[44] Zijie Zeng, Jing Lin, Lin Li, Weike Pan, and Zhong Ming. 2020. Next-Item recommendation via collaborative filtering with bidirectional item similarity. *ACM Transactions on Information Systems* 38, 1 (2020), 7:1–7:22. DOI: https://doi.org/10.1145/3366172

[45] Shuai Zhang, Yi Tay, Lina Yao, Aixin Sun, and Jake An. 2019. Next item recommendation with self-attentive metric learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press.

[46] Yuchen Zhang, Amr Ahmed, Vanja Josifovski, and Alexander J. Smola. 2014. Taxonomy discovery for personalized recommendation. In *Proceedings of the ACM International Conference on Web Search and Data Mining*. ACM, NY, 243–252. DOI: https://doi.org/10.1145/2556195.2556236

[47] Hongke Zhao, Qi Liu, Hengshu Zhu, Yong Ge, Enhong Chen, Yan Zhu, and Junping Du. 2018. A sequential approach to market state modeling and analysis in online p2p lending. *IEEE Transactions on Systems, Man and Cybernetics: Systems* 48, 1 (2018), 21–33.

[48] Nengjun Zhu and Jian Cao. 2019. CPL: A combined framework of pointwise prediction and learning to rank for top-N recommendations with implicit feedback. In *Proceedings of the International Conference on Web Information Systems Engineering*. Springer, 259–273. DOI: https://doi.org/10.1007/978-3-030-34223-4_17

[49] Nengjun Zhu, Jian Cao, Yanchi Liu, Yang Yang, Haochao Ying, and Hui Xiong. 2020. Sequential modeling of hierarchical user intention and preference for next-item recommendation. In *Proceedings of the International Conference on Web Search and Data Mining*. ACM, 807–815. DOI: https://doi.org/10.1145/3336191.3371840

[50] Qianna Zhu, Zeliang Song Xiaofei Zhou, and Li Guo Jianlong Tan. 2019. DAN: Deep attention neural network for news recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press, 5973–5980. DOI: https://doi.org/10.1609/aaai.v33i01.33015973

[51] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to do next: Modeling user behaviors by Time-LSTM. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 3602–3608. DOI: https://doi.org/10.24963/ijcai.2017/504

Received August 2020; revised May 2021; accepted July 2021