Entity Recommendation with Negative Feedback Memory Networks for Topic-oriented Knowledge Graph Exploration

Yi Yang
Institute of Automation, Chinese Academy of Sciences
Beijing, China, yangyi@ia.ac.cn

Meng Li
Institute of Automation, Chinese Academy of Sciences,
Beijing, China, meng_li@ia.ac.cn

Jian Wang*
Institute of Automation, Chinese Academy of Sciences
Beijing, China, jian.wang@ia.ac.cn

Weixing Huang
Institute of Automation, Chinese Academy of Sciences,
Beijing, China, weixing.huang@ia.ac.cn

Yun Wang
Institute of Automation, Chinese Academy of Sciences,
Beijing, China, y.wang@ia.ac.cn

*Corresponding author: Jian Wang

***Abstract*—Knowledge Graph Exploration is an interactive knowledge discovery process over the knowledge graph. Entity recommendation deals with the information overflow issue when exploring the large-scale unfamiliar knowledge graphs. The traditional personalized entity recommendation methods for knowledge graph explorations rarely consider the adaptive topic-oriented long-term positive- and negative intent modelling. In this paper, we propose a topic-oriented entity recommendation method during the knowledge graph exploration. We build a Negative Feedback Memory Network model for obtaining the user's long-term negative intents. We propose a Transformer-based sequence encoder for the positive intents. We dynamically obtain the adaptive intents by aggregating the positive- and negative intents by the proposed Intent Attention mechanism. Experiments show that our method has advantages in TopK entity recommendations.**

Keywords- entity recommendation, knowledge graph, negative feedback, memory network, knowledge graph exploration.

I. INTRODUCTION

Nowadays, Knowledge Graphs (KGs) are widely used for introducing features of knowledge facts to enhance intelligent analysis and machine learning methods in many fields, e.g., data mining, recommender systems, fault reasoning, and reliability engineering. Knowledge Graph Exploration (KGE) [1] [2] is a knowledge discovery process over a KG. Taking Question Answering (QA) systems as an example, a conversation can be treated as a walk over a KG. The QA pairs are related to the corresponding entities of the KG [3] in this scenario. Another example is visual analytics, on which the user visually explores insights step-by-step over a KG with the help of algorithms [4].

When exploring a large-scale unfamiliar KG, it is difficult for the user to effectively find the interesting entities. This is the information overflow problem of KGE. To address this issue, many studies worked on Personalized Entity Recommendation (PER) that can suggest contextually relevant entities of a KG for the given entities. PER for KGE is usually modelled as a prediction of the most valuable KG entities according to the given entities. The key of PER is to model the user's dynamic intents or desires along with the process of KGE [5].

Previous studies mainly focus on the models based on the positive user behaviors [6], also known as the positive feedbacks, which explicitly or implicitly show the interests. The negative feedbacks, particularly the long-term negative feedbacks, were rarely considered for modelling the user's intents. Besides, the previous methods rarely take the centrality of the session entities into account. During the process of KGE, the user usually focuses on a serial of centric topics. Each topic contains a few entities that form a tree-like subgraph of a KG. The topic represents the aggregation of strongly related entities, which is beneficial for PER. In addition, the previous work learns a fixed embedding of the input data to rank the candidate entities without considering the semantic relations between each input entity and candidate entities. These relations will improve the representation of the exploration intents in KGE.

In this paper, for addressing the critical issues, we propose a PER method for KGE with the following contributions: (1) we model the long-term negative feedbacks with memory networks for representing the negative intents; (2) we propose the topic-oriented graph structures for better modelling the positive- and negative intents; (3) we introduce intent attention mechanism for obtaining the adaptive positive- and negative intents; (4) we perform experiments to evaluate the benefits of our method.

The rest of the paper is organized as follows. Section 2 depicts the related work. Section 3 defines the problems and Section 4 describes the proposed method. Section 5 and Section 6 depict the experiments for evaluating our method. Finally, conclusion and future work are given in Section 7.

II. RELATED WORK

A. Knowledge Graph Exploration

KGE is the gradual discovery and content understanding of a large-scale unfamiliar KG. Lissandrini et al. [1] [2] defined KGE as the computer-assisted interactive process of progressive analysis of a KG. KGE is a critical study for human-oriented tasks and machine-oriented tasks. For the human-oriented tasks, the basic goal of KGE study is to help the user to quickly understand the unfamiliar KG for finding valuable insights. For

the machine-oriented tasks, the KGE study can enhance the prediction algorithms in the fields of QA systems, personalized search, recommendation, and KG-based data mining. An exploratory behavior of KGE is a user action for investigating a desired entity. Shi et al. [7] proposed a relevant-path-set-based method to search targets by estimating the path cost of exploratory search. Kuric et al. [8] compared different methods of KGE for estimating the usability of the SparQL query builder. Medlar and Głowacka elaborated [9] that the exploratory focuses on the knowledge discovery and planning. It differs from the traditional lookup search. Zheng et al. [10] proposed a random walk-based diversity-aware entity exploration method that represents user's conceptual exploration with concept graph for entity exploration over a corresponding KG. Tawil et al. [11] proposed a method that facilitates users' explorations using data graphs by increasing the users' domain knowledge.

B. Recommender System

Recommender System (RS) [12] plays an important role for personalized services in many fields [5] [13] [14] [15] [16] [17] [18] [19] [20].Session-based Recommender System (SBRS) [21] predicts items that the user may be interested in according to a session of user-item interactions. SBRS learns the session features by modelling the sequential dependencies. A closely relevant study to SBRS is Sequential Recommender System [22]. SBRS mainly works on the next-item recommendations. It is valuable for the anonymous session data. GRU4Rec [23] applies the traditional sequential model GRU to extract the sequential features. BERT4Rec [24] applies the sophisticated language model BERT to model the session data by making full use of the attention mechanism. SR-GNN [25] is a well-known method that uses Graph Neural Network to model the session graph by taking the relations among multiple sessions into account.

C. Personalized Entity Recommendation

PER aims at the entity suggestions over a KG or a semantic web based on the user behaviors. Entity2rec [26] models the user-item relations for recommendations by using property-specific KG embeddings. Ni et al. [27] proposed a layered-graph based embedding approach for the Wikipedia-relevant PER. Huang et al. [28] proposed a multi-task learning method that uses deep neural networks to jointly learn both the PER task and document ranking task for optimizing the web search. Besides, a similar study is entity set expansion that predicts the relevant knowledge entities by analyzing semantic relations between the given seed entities and the candidate entities. Zhang et al. [29] proposed an entity set expansion method that can generate candidate class names for predicting the related entities applying a pre-trained language model. Set-CoExpan [30] generates auxiliary sets using the corpus-based model for capturing better semantic relations between entities. BootstrapNet [31] uses a deep learning method for entity prediction, which represents the bootstrapping process by an encoder-decoder model. Yu et al. [32] proposed a course concept expansion framework for MOOC by interactive games and external knowledge.

D. Memory Network

Memory network [33] is a framework for managing the long-term historical data using external memory in order to extract long-distance dependent features for sequential-based tasks. Key-Value Memory Networks [34] use the key-value structure for efficiently storing the large-scale textual data. The knowledge-enhanced sequential recommendation [35] combines RNN and key-value memory networks for improving sequential recommendations. KEPS [36] is a KG-enhanced personalized search method that models long-term entity preference using key-value memory networks. RUM [37] integrates collaborative filtering with memory networks for sequential recommendations. Wu et al. [38] proposed the Feedback Memory Network (FMN) for modelling user's feedback behaviors for query suggestion during the search process. Zhou et al. [39] introduced a memory networks-based method to model the complex re-finding behaviors for personalized search.

E. Attention Mechanism

Attention mechanism can effectively extract the important features by assign weights to features based on their importance to the targets. The classic Attention mechanism [40] is used in the encoder-decoder framework for machine translation. Self-Attention mechanism is a well-known variant applied in the sophisticated models Transformer [41] and BERT [42] for modelling the sequence data. The Attention mechanism is also widely applied in recommender systems [43] [44] [45].

III. PROBLEM STATEMENT

A. Preliminaries

A Knowledge Graph is a heterogeneous graph structure with semantic properties, which is denoted by $KG = (E, R)$, where E is the entity set and $R: E \times E$ represents the relation set. The KG is modelled with the triple $< Subject, Predicate, Object >$, where $Subject, Object \in E$ and $Predicate \in R$.

A Knowledge Graph Exploration is defined as a path walk process whose result is a directed graph denoted as a directed graph by $KGE = (E^{KGE}, R^{KGE}, O^{KGE})$, where the $E^{KGE} \subseteq E$ represents entity set, R^{KGE} represents relation set, and O^{KGE} indicates an order set of relations. Let $\prec$ be an exploration order of relations, the order set is denoted by $O^{KGE} = (R^{KGE}, \prec)$. The relation is denoted by $r_{i,j} = (e_i, e_j) \in R^{KGE}$, where entities $e_i, e_j \in E^{KGE}$ with the direction from e_i to e_j. For the KGE, a session is defined as a sequence of KG entities.

In this paper, topic is defined as a set of KG entities that directly connect with the same entity among them. The topic represents semantic meaning of several exploratory behaviors. A topic consists of a central entity (i.e., topic core) and its direct neighbor entities on a KG (i.e., topic context). Topic-oriented Knowledge Graph Exploration (ToKGE) is a special KGE that can be divided into a sequence of topics. During the process of ToKGE, it explores the entities of a topic over the KG before moving to the next topic core entity. Fig. 1 shows an example. The ToKGE process is a session containing 2 topics. The user explores first *The Oscars*-centric topic, and then the *Forrest Gump*-centric topic. The first topic contains a topic core entity *The Oscars* and two topic context entities *Award Category* and *Forrest Gump*. The exploration of the both entities is treated as the exploration inside *The Oscars* -centric topic. After that, the users explores the entities *Tom Hanks* and *Robert Zemeckis* around the new topic core entity *Forrest Gump*, i.e., the exploration topic is changed. ToKGE is denoted by $ToKGE = (KGE, T)$, where $T = (t_1, t_2, \dots, t_m)$ indicates a topic set. The

structure of a topic t is a subgraph of KGE, denoted by $t_i = \left(E_{t_i}^{KGE}, R_{t_i}^{KGE}\right) \in T$, where $E_{t_i}^{KGE} \subseteq E^{KGE}$ and $R_{t_i}^{KGE} \subseteq R^{KGE}$.

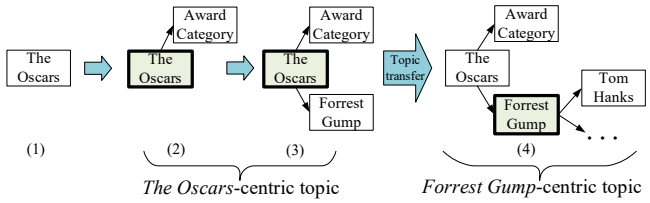


Fig. 1 Process of ToKGE. (1) the given entity; (2) and (3) show the exploration of the neighbour entities in the range of *The Oscars*-centric topic; (4) shows the exploration in the range of *Forrest Gump*-centric topic.

B. Task definition

When the user explores over a KG, the KG is treated as the context KG of the explored entity session. For a new topic core entity, the task of PER of ToKGE is to find k topic context entities that are most likely interested in by the user and do not appear in the session. For quantifying the user's interest, we need to define an interest function $f(session, topic\_core, KG)$ to estimate the interest for each candidate entity of KG. With the interest estimation, the candidate entities will be ranked for obtaining the k most valuable entities. The process of PER is shown in Fig. 2. Based on a given historical session with the last explored entity *The Oscars*, PER finds three possible interested entities as candidates over the context KG. Next, the user explores the interesting entities *Award Category* and *Forrest Gump*. Then, the user has no more interests of current topic and transfers interest to the *Forrest Gump*-centric topic. A new PER will be requested for suggesting possible interested entities for *Forrest Gump*-centric topic.

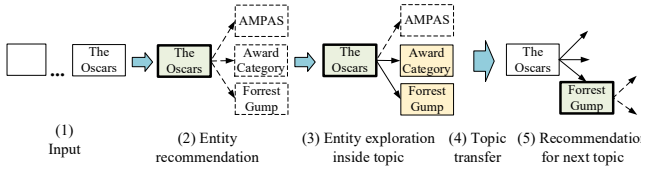


Fig. 2 Process of Personalized Entity Recommendation. (1) historical session data; (2) PER; (3) Entity exploration inside topic: exploring the entities *Award Category* and *Forrest Gump*; (4) Topic transfer; (5) PER for next topic.

The user's feedback of PER is essentially the response to the recommendation results. The feedback can be categorized into positive feedback and negative feedback that respectively represent the like and dislike, implicitly. We define the indicator function as Eq. (1).

$$fd(e_i) = \begin{cases} positive, & if\ e_i\ is\ explored \\ negative, & otherwise \end{cases} \tag{1}$$

where e_i indicates an entity.

A sequence of entities contained by a session are considered as positive feedbacks. In this paper, we define the negative feedbacks as a part of the recommended entities that the user is not interested in. For example, in Fig. 2 (3), *Award Category* and *Forrest Gump* are considered as a positive feedback entities and *AMPAS* is treated as a negative feedback entity.

IV. METHOD

A. Concept

In this paper, there are three types of critical features: positive feedbacks with the exploration order, knowledge features, and negative feedbacks. We build different types of structures based on the session to respectively learn the features as shown in Fig. 3: session sequence is a sequence-based structure according to the orders of the explored entities; session graph is a graph-based structure for modelling the negative feedbacks. The session graph can be divided in to a Chain-structured Subgraph (CSG) representing transfers of the topics and one or more Tree-structured Subgraphs (TSGs) respectively representing the topic contents. CSG represents the topic transfer and TSG represents the topic content. In the example, the session contains three topics: software company-centric, Apple-centric, and iPhone-centric. A topic core can be a topic context of the last topic when the explored entities are continuous.

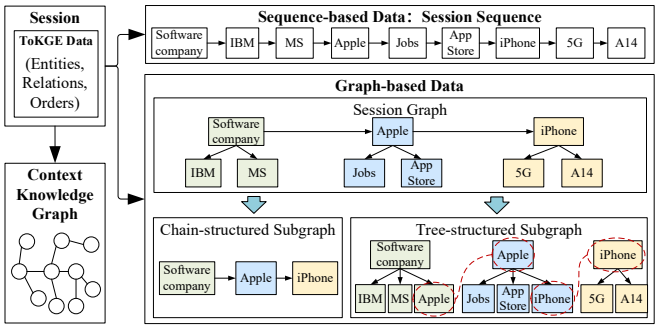


Fig. 3 Structure construction of session data.

We propose metrics for evaluating the CSG and TSG.

- Size of a CSG (CSG Size): for a session with n topics

$$size_{CSG} = \sum_{i=1}^{n} 1_A(entity_i) \tag{2}$$

with an indicator function :

$$1_A(entity) = \begin{cases} 1, & if\ entity\ is\ topic\ core \\ 0, & if\ entity\ is\ topic\ context \end{cases} \tag{3}$$

- Mean Size of CSGs: for m sessions in total

$$mean\ size_{CSGs} = \frac{1}{m}\sum_{i=1}^{m} size_{CSG_i} \tag{4}$$

- Size of a TSG (TSG size): for a topic with l entities

$$size_{TSG} = l - 1 \tag{5}$$

- Average Size of TSGs: for a session with n topics

$$avg.size_{TSGs} = \frac{1}{n}\sum_{i=1}^{n} size_{TSG_i} \tag{6}$$

- Mean Average Size of TSGs: for m sessions in total

$$mean\ avg.size_{TSGs} = \frac{1}{m}\sum_{i=1}^{m} avg.size_{TSG_i} \tag{7}$$

Algorithm 1 depicts the construction process of the session graph according to a session and a context KG. For a session entity e_i, the algorithm updates the entity set and the relation set

of the session graph (line 1-4). The topic transfer will happen under either of two conditions. One is that e_i^{SE} does not have direct connection with the current topic core entity over KG (line 5-6); another is that e_i^{SE} has a direct connection with the session entity e_{i+1}^{SE} over KG (7-8). The current topic core entity will be updated with e_i^{SE} for the topic transfer. Finally, the algorithm outputs a session graph.

Algorithm 1: Session Graph Construction

Input: $SE = [e_1^{SE}, e_2^{SE}, \dots, e_n^{SE}]$, $KG = (E^{KG}, R^{KG})$ with $r_{i,j} = (e_i^{SE}, e_j^{SE}) \in R^{KG}$ and $e_i^{SE}, e_j^{SE} \in E^{KG}$
Output: directed graph $SG = (E^{SG}, R^{SG})$
1: Initialize $E^{SG} \leftarrow \emptyset$, $R^{SG} \leftarrow \emptyset$
2: **for** $i = 1$ to n **do**
3: $E^{SG} \leftarrow E^{SG} \cup \{e_i^{SE}\}$
4: $R^{SG} \leftarrow R^{SG} \cup \{(e_{core}, e_i^{SE})\}$
5: **if** not exist $r_{core,i} \in R^{KG}$ **then**
6: $e_{core} \leftarrow e_i^{SE}$
7: **else if** exist $r_{i,i+1} \in R^{KG}$ **then**
8: $e_{core} \leftarrow e_i^{SE}$
9: **end if**
10: **end for**
11: **return** *SG*

B. Architecture

The architecture of our method consists of three important parts for learning the features of the user's intent. The Session Interest layer models the positive intents and the Negative Feedback Memory Networks layer learns the negative intents. The Aggregation layer integrates the positive intent, the negative intent, and the knowledge feature for each candidate entities. We obtain the recommended entities by similarity computing and the softmax function. The architecture is shown in Fig. 6.

C. Preparation

In this paper, the symbol $\hat{e}_i$ denotes the embedding of an entity e_i, an embedding set is indicated as $\hat{e} = \{\hat{e}_1, \hat{e}_2, \dots, \hat{e}_{|e|}\}$. We use the knowledge graph embedding model *TransE* [46] to encode the entities of the context KG by learning the semantic distance between entities. The knowledge semantic of the entities can be denoted Eq. (8).

$$\hat{e}^{KG} = TransE(KG) = \{\hat{e}_1^{KG}, \hat{e}_2^{KG}, \dots, \hat{e}_{|KG|}^{KG}\} \quad (8)$$

where $\hat{e}^{KG} \in \mathbb{R}^{d_{KG} \times |KG|}$ consists of entity embeddings. The candidate entity set is denoted as $CE = \{e_1^{CE}, e_2^{CE}, \dots, e_m^{CE}\}$ that contains the direct neighbour entities of a specified topic core entity over the context KG, which do not appear in the session.

D. Session Interest Layer

1) Session Sequence: A session sequence contains a set of entities, which is denoted by $SE = [e_1^{SE}, e_2^{SE}, \dots, e_n^{SE}]$, where e_i^{SE} is an entity belonging to the context KG.

2) Input Encoder: Inspired by the architecture of BERT, we encode the session sequence by aggregating knowledge feature, the topic feature, and the position feature as shown in Fig. 4.The session entity $\hat{e}_i^{SE}$ is encoded as Eq. (9) and Eq.(10):

$$\hat{e}_i^{SE} = \hat{e}_i^{KG} + \hat{e}_i^{top} + \hat{e}_i^{pos} \quad (9)$$

$$\hat{e}_i^{top} = W^{top} H_i^{top}, \quad \hat{e}_i^{pos} = W^{pos} H_i^{pos} \quad (10)$$

where $\hat{e}_i^{SE} \in \mathbb{R}^{d_{KG}}$; $\hat{e}_i^{top} \in \mathbb{R}^{d_{KG}}$ encodes the topic including e_i^{SE}, where the $H_i^{top} \in \mathbb{Z}^t$ denotes the one-hot vector of topic and $W^{top} \in \mathbb{R}^{d_{KG} \times t}$. The entities contained by the identical topic have the same topic embedding; $\hat{e}_i^{pos} \in \mathbb{R}^{d_{KG}}$ denotes the position of e_i^{SE}, where the $H_i^{pos} \in \mathbb{Z}^g$ denotes the one-hot vector of position and $W^{pos} \in \mathbb{R}^{d_{KG} \times g}$. The session entity set is denoted as $\hat{e}^{SE} = \{\hat{e}_1^{SE}, \hat{e}_2^{SE}, \dots, \hat{e}_n^{SE}\}$.

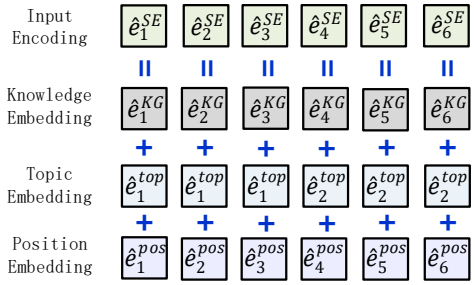


Fig. 4 Input of sequence encoder

3) Sequence Encoder: In this paper, we build a sequence encoder consisting of three Transformer encoder blocks. Fig. 5 shows the structure of the Transformer encoder block that contains several important units. The Multi-Head Self-Attention aggregates multiple attention heads that respectively represent subspaces at different positions for enhancing the feature representations, which is defined as Eq. (11).

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h) W^O \quad (11)$$

where $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$, $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$, and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$ are learnable parameters, h is the number of attention heads, d_{model} denotes dimension of embedding. In this paper, we build the model with $h = 4$. The Q, K, and V are projections of the session entity set. We define $Q = (\hat{e}^{SE})^{\mathrm{T}} W_Q$, $\hat{e}^{SE} \in \mathbb{R}^{d_{model} \times n}$, where $W_Q \in \mathbb{R}^{d_{model} \times d_{model}}$; $K = (\hat{e}^{SE})^{\mathrm{T}} W_K$, where $W_K \in \mathbb{R}^{d_{model} \times d_{model}}$; $V = (\hat{e}^{SE})^{\mathrm{T}} W_V$, where $W_V \in \mathbb{R}^{d_{model} \times d_{model}}$. The encoder uses the scaled dot product attention defined as in Intent Attention mechanism. The Add and Norm layer works on aggregating and normalizing a group of embeddings. The Position-wise Feed-Forward Network contains two-layer fully connected networks with a ReLU activation in between: $FFN = MLP_2(ReLU(MLP_1))$. The output of the sequence encoder is a set of entity embeddings $\hat{e}^{seq} = \{\hat{e}_1^{seq}, \hat{e}_2^{seq}, \dots, \hat{e}_n^{seq}\} \in \mathbb{R}^{d_{model} \times n}$.

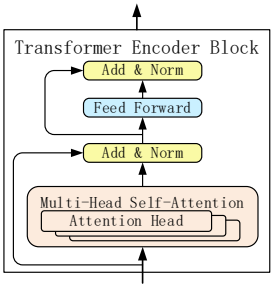


Fig. 5 Transformer encoder block.

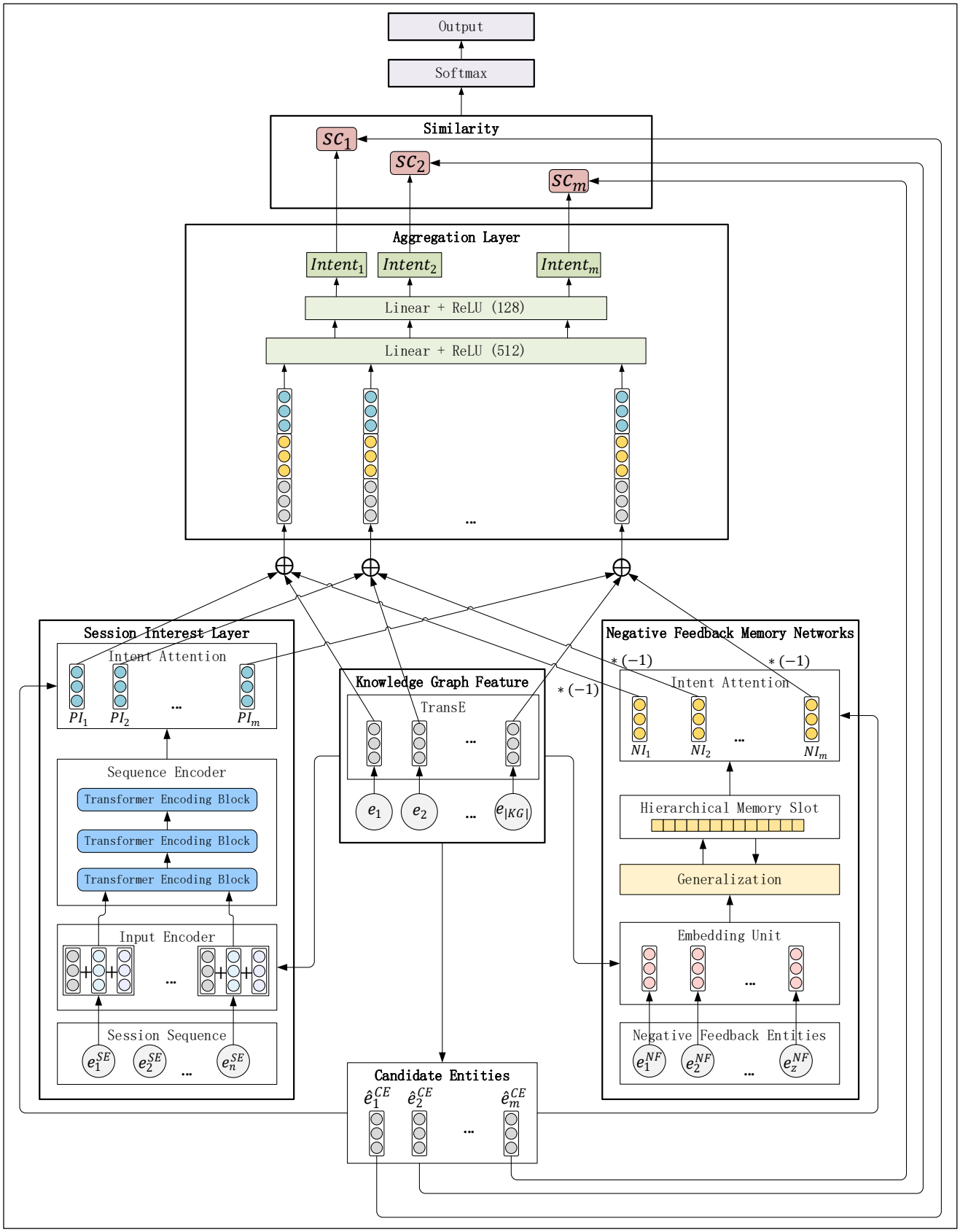


Fig. 6 Architecture

4) Positive Intents: Using the Intent Attention mechanism, we obtain the weighted embeddings based on the importance to the candidate entities, which is defined as $\hat{e}^{PI} = IA(\hat{e}^{seq}, CE)$. The positive intent set is defined as Eq. (12) .

$$PI = \{\hat{e}_1^{PI}, \hat{e}_2^{PI}, \dots, \hat{e}_m^{PI}\} \tag{12}$$

where $PI \in \mathbb{R}^{d_{model} \times n}$.

E. Negative Feedback Memory Network (NFMN)

1) Embedding Unit: Corresponding to l topics, the negative feedback entity set e^{NF} is divided into l groups: $e_{top}^{NF} = \{e_{top_1}^{NF}, e_{top_2}^{NF}, \dots, e_{top_l}^{NF}\}$, where $e_{top_i}^{NF} = \{e_{i,1}^{NF}, e_{i,2}^{NF}, \dots, e_{i,p}^{NF}\}$ and $e_{i,j}^{NF} \in e^{NF}$. The initial value of a negative feedback entity is the knowledge embedding of the entity. The embedding unit has an embedding lookup function that converts the input values to new embeddings. The encoding is defined as Eq. (13).

$$\hat{e}^{NF} = W_{in}^{\mathrm{T}} \hat{e}_{nf}^{KG} \tag{13}$$

where $\hat{e}_{nf}^{KG}$ denotes the embeddings of negative entities of the context KG, and $W_{in} \in \mathbb{R}^{d_{KG} \times q}$ is the learnable parameters.

2) Hierarchical Memory Slot: We define the memory of NFMN as a First-In First-Out (FIFO) queue. We propose a Hierarchical Memory Slot (HMS) in order to deal with both the topic orders and the negative feedback entities of each topic. It is defined as Eq. (14).

$$HMS = \left(\hat{e}_{top}^{NF}, O_{top}\right) \tag{14}$$

where $O_{top} = \left(\left(\hat{e}_{top_i}^{NF}, \hat{e}_{top_j}^{NF}\right), \prec\right)$ indicates the orders of the memory elements. HMS contains topic hierarchy and entity hierarchy, which is shown in Fig. 7. The topic hierarchy contains a set of memory slots for storing the sequential memory elements that represent topics. In the entity hierarchy, each memory element contains a few negative feedback entities of a topic. We define a metric *NF length* as the number of obtained topics that are read from the memories. For example, in Fig. 3, the NF length is 3 because NFMN will use the negative entities regarding the *Software Company*-, *Apple*-, and *iPhone*-centric topics for encoding negative intents.

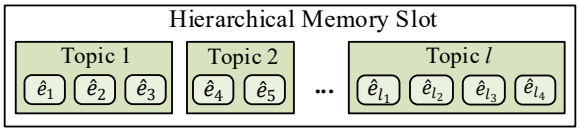


Fig. 7 HMS contains l memory slots for l topics. Each slot includes negative feedback entities of the corresponding topic.

3) Generalization: The Generalization unit manages the memorized data. When updating memories, for the given input regarding l topics, the Generalization unit updates the memory slots storing the earliest l topics with the input data; when reading memories, the Generalization unit obtains the last l memory based on the update order.

4）Negative Intents: We compute the negative intent attention values by the Intent Attention mechanism according to the importance between the negative feedback entities and each candidate entities. The output is defined as $\hat{e}^{NI} = IA(\hat{e}^{NF}, CE)$. The negative intent set is obtained by $(-1) \times \hat{e}^{NI}$, which is defined as Eq. (15).

$$NI = \{(-\hat{e}_1^{NI}), (-\hat{e}_2^{NI}), \dots, (-\hat{e}_m^{NI})\} \tag{15}$$

where $NI \in \mathbb{R}^{d_{model} \times z}$.

F. Aggregation Layer

1) Aggregation: We concatenate the positive intent, the negative intent, and the knowledge embedding. The user's intents can be obtained by linear layers with ReLU activations defined as Eq. (16) and Eq. (17).

$$I_i = PI_i \oplus NI_i \oplus \hat{e}_i^{KG} \tag{16}$$

$$Intent_i = ReLU\left(W_2^{\mathrm{T}} ReLU\left(W_1^{\mathrm{T}} I_i + b_1\right) + b_2\right) \tag{17}$$

where $I_i \in \mathbb{R}^{2d_{model}}$, $W_1 \in \mathbb{R}^{2d_{model} \times U}$, and $W_2 \in \mathbb{R}^{2d_{model} \times V}$. We set $U = 512$ and $V = 128$ in this paper.

2) Loss Function: After obtaining the intents, we compute the score between each intent and the corresponding candidate entity as Eq. (18).

$$sc_i = Intent_i^{\mathrm{T}} \hat{e}_i^{CE} \tag{18}$$

Then, the scores are normalized by softmax function as Eq. (19).

$$y' = \mathrm{softmax}(sc) \tag{19}$$

where $sc \in \mathbb{R}^m$ denotes the scores for m candidate entities, $y' \in \mathbb{R}^m$ indicates the probabilities of recommendations. We define the loss function as the binary cross-entropy of the prediction and the ground truth y_i as Eq. (20).

$$\mathcal{L} = -\sum_{i=1}^{m} y_i log\left(\sigma(y_i')\right) + (1 - y_i) log\left(\sigma(1 - y_i')\right) \tag{20}$$

G. Intent Attention

For a given session of source entities and a target entity, the Intent Attention mechanism encodes the session by considering the attentions between session entities and the target entity, which is shown in Fig. 8. With the Intent Attention, the fine-grained intent features can be dynamically obtained regarding the importance to each candidate entity.

For different target entities, the Intent Attention mechanism provides different session encodings rather than a fixed value in order to adaptively represent the importance to the target entity. First, we define the similarity score between source entity i and target entity j using the scaled dot product defined as Eq. (21).

$$score_{i,j} = \frac{1}{\sqrt{d}}\left((\hat{e}_i^{source})^{\mathrm{T}} \hat{e}_j^{target}\right) \tag{21}$$

where d indicates embedding dimension. The normalized score $a_{i,j}$ is defined as Eq. (22).

$$a_{i,j} = \mathrm{softmax}\left(score_{i,j}\right) = \frac{exp\left(score_{i,j}\right)}{\sum_{i=1}^{n} exp\left(score_{i,j}\right)} \tag{22}$$

The intent attention IA_j for target entity j is treated as a normalized weighted embedding of each source entity, which is defined as Eq. (23).

$$IA_j = \sum_{i=1}^{N} a_{i,j} \hat{e}_i^{source} \tag{23}$$

where N is the number of the source entities. The intent attention set is denoted as $IA = \{IA_1, IA_2, \dots, IA_m\}$ that contains intent attentions respectively for m target entities. The Intent Attention mechanism is respectively used for computing the positive intents and the negative intents.

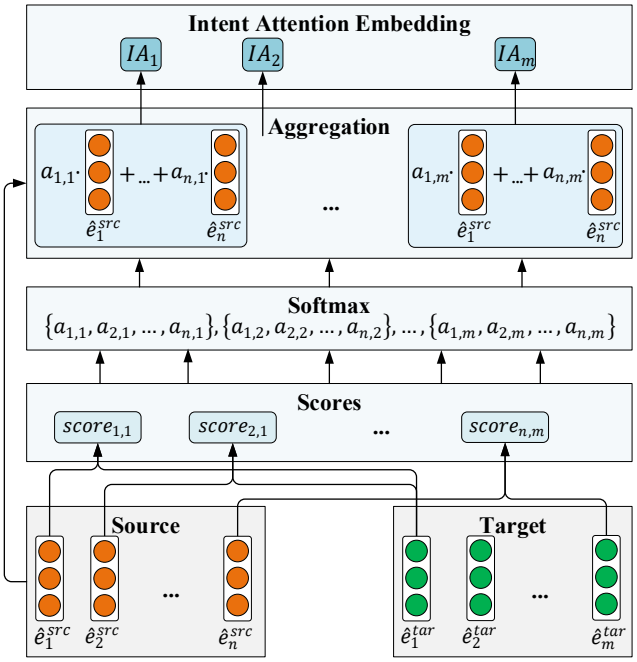


Fig. 8 Intent Attention

H. Model Learning

The entity embeddings of the context knowledge graph need to be learned before training the model. These embeddings will not be updated in the training process. The positive intents and the negative intents can be computed in parallel.

1) Training: The entities extracted with the parameters of the CSG size and the NF length are used for model training. The parameters represent the number of topics respectively for computing positive intents and negative intents. Algorithm 2 demonstrates the learning process of our model. The input includes a session of entities, candidate entities, a context KG, the CSG size and the NF length. The learnable parameters and entity embeddings of the context KG need to be initially processed (line 1-2). We construct the inputs by aggregating the features of topic, position, and knowledge (line 4-8). The algorithm computes the sequence embeddings of entities using the Sequence Encoder and obtains the positive intents by the Intent Attention mechanism (line 9-12). Meanwhile, we extract the negative feedback entities and compute the negative intents basically using NFMN and the Intent Attention mechanism (line 13-20). For each candidate entity, its positive intent, negative intent, and the knowledge embedding are concatenated (line 19). Then, we compute the loss and update learnable parameters by stochastic gradient descent (line 21-30). With a number of iterations, the algorithm outputs the recommendation function with the learned parameters.

Algorithm 2: Learning Algorithm

Input: SE, CE, KG, CSG size cs, NF length nfl
Output: rec. function $\mathcal{F}(e|\theta)$ with parameters θ

1: Initialize all parameters: init(θ)
2: $\hat{e}^{KG} \leftarrow TransE(KG)$
3: **for** number of training iteration **do**
4: SE $\leftarrow$ SEextract(SE, cs)
5: **for** $i = 1$ to n **do**
6: $\hat{e}_i^{top} = W^{top} H_i^{top}$ and $\hat{e}_i^{pos} = W^{pos} H_i^{pos}$
7: $\hat{e}_i^{SE} = \hat{e}_i^{KG} + \hat{e}_i^{top} + \hat{e}_i^{pos}$
8: **end for**
9: **for** 3 iterations **do**
10: $\hat{e}^{seq} \leftarrow$ Sequence Encoder$(\hat{e}^{SE})$
11: **end for**
12: $\hat{e}^{PIA} \leftarrow IA(\hat{e}^{seq}, CE)$
13: $\hat{e}_{nf}^{KG} = NFextract(SE, KG, nfl)$
14: $\hat{e}^{NF} = W_{in}^{\mathrm{T}} \hat{e}_{nf}^{KG}$
15: Update the Hierarchical Memory Slots.
16: $\hat{e}^{NIA} \leftarrow IA(\hat{e}^{NF}, CE)$
17: **for** $i = 1$ to m **do**
18: $\hat{e}_i^{NIA} = (-1) \times \hat{e}_i^{NIA}$
19: $I_i = \hat{e}_i^{PIA} \oplus \hat{e}_i^{NIA} \oplus \hat{e}_i^{KG}$
20: **end for**
21: $Intent = ReLU\left(W_2{}^{\mathrm{T}} ReLU\left(W_1{}^{\mathrm{T}} I + b_1\right) + b_2\right), I_i \in I$
22: **for** $i = 1$ to m **do**
23: $sc_i = Intent_i^{\mathrm{T}} \hat{e}_i^{CE}$
24: **end for**
25: $y' = \mathrm{softmax}(sc)$
26: $\mathcal{L} = -\sum_{i=1}^{m} y_i log\left(\sigma(y_i')\right) + (1 - y_i) log\left(\sigma(1 - y_i')\right)$
27: Compute gradients for each learnable parameter $\partial\mathcal{L}/\partial\theta_i$, $\theta_i \in \theta$
28: Update θ by stochastic gradient descent with learning rate η
29: **end for**
30: **return** recommendation function $\mathcal{F}(e|\theta)$

2) Test: The proposed model is used to predict the next K session entities for the TopK recommendation. In this paper, we do not evaluate the exact orders of the recommended entities.

V. Experimental Setup

A. Research Questions

We evaluate the performance of our method for the following research questions:

- RQ1: Can our method outperforms the state-of-the-art methods for the PER tasks regarding ToKGE?
- RQ2: How does our method performs on the session data with different settings?
- RQ3: How is the contributions of each critical component in our method?

According to these research questions, we select databases, metrics, and baseline methods for the experiments.

B. Datasets

We build the experimental data based on QA-datasets because the QA-datasets naturally contain quite a few session data for forming CSGs and the TSGs. The experimental data is shown in TABLE I.

TABLE I STATISTICS OF EXPERIMENTAL DATASETS

	KdConv-based dataset	CSQA-based dataset
# Entities	36,052	182,930
# Relations	36,617	183,641
Total sessions	830	1,426
Mean size of CSGs	3.201	9.728
Mean avg. size of TSGs	5.068	2.134

1) KdConv-based Dataset (KdConv-bd): We construct the dataset based on KdConv [3] that is a dataset for multi-domain knowledge-driven conversation. KdConv contains 4,500 dialogs with an average number of 19 turns in terms of film, music, and travel. The entities extracted from the dialogs are respectively mapped to the entities of DBpedia [47]. It forms a knowledge graphs that contains 13,072 entities and 9,115 relations. The experiments need a medium number of direct neighbour entities for a given entity of tree-structured data in order to ensure the experimental methods have enough neighbours as candidates for the TopK recommendations. Therefore, we sample the session data with the CSG size $\geq$3 and average TSG size $\geq$ 2. In order to generate the negative feedback data, we extend the samples with the existing KGs including DBpedia and Ownthink [48]. For each topic core entity of CSG, we identify the corresponding entity of the KGs. Then, we look up the direct neighbour entities and relations of the KG entities. Then, we eliminate duplicates according to the synonyms directory. Last, we randomly select a certain number of neighbour entities to supplement each topic that totally contains not more than 15 entities. We obtain a dataset containing 830 sessions involving 36,052 entities and 36,617 relations. The mean size of CSG is 3.201 and mean average size of TSG is 5.068.

2) CSQA-based dataset (CSQA-bd): For the experiments, we use the dataset CSQA [49] that consists of a large number of sequential complex questions. CSQA has more than 150K dialogs that are labelled with the corresponding entities and relations based on Wikidata. CSQA is suitable for this paper because it can be easily transformed into graphs with CSGs and TSGs. We only use the simple QA pairs of CSQA in order to well match our experiments. The statistical QA pairs for clarification and count are filtered out because the results of the pairs are numbers rather than entities. We sample the data with the CSG size $\geq$ 9 and the average TSG size $\geq$ 2. The sampled data has the mean size of CSG of 9.728 and the mean average size of TSG of 2.134. In order to generate the negative feedback data, we extend CSQA with Wikidata. For each entity of CSQA, we first find the corresponding entity in Wikidata. Then, we find its direct neighbor entities and relations that do not appear in CSQA. Finally, we add the identified entities and relations into CSQA. The extended dataset is CSQA-bd.

C. Baselines

In the experiments, we compare our method with eight representative and state-of-the-art methods that are selected from three most related areas.

1) Conventional Methods: traditional and widely used methods.

- PageRank [50] is a random walk-based graph ranking algorithm. It can predict graph nodes according to their importance to the given entities. PageRank is basically works on directed graphs. Therefore, we treat the undirected experimental graph as a bi-directed graph for being made available for PageRank. In the experiments, we train PageRank with the experimental KG for computing the structural features of the KG. After training, for a specified entity, PageRank outputs a set of ranked entities. This algorithm represents the traditional graph analysis methods that do not take the dynamic changes of graphs into account.
- Graph Attention Networks (GAT) [51] is a well-known Graph Neural Network model that introduces the multi-head attention mechanism into the propagation step for learning the relative weights between the linked nodes in a graph. It computes the hidden states of each nodes for the weighting the relations between the nodes. We select GAT because it is the representative of the conventional graph deep learning methods. In the experiments, GAT learns the entity embeddings by computing the given knowledge graph. Based on the embedding similarity of each pair of the specified entity and its direct neighbours, GAT can rank the neighbours for the TopK entity recommendations.
- Sequence-to-Sequence (Seq2Seq) [52] is an Encoder-Decoder model integrating with RNN models. Seq2Seq is the widely used deep learning model for analysing the sequence data. The model can predict a sequence of objects according to a list of given ordered objects. In this paper, we model the experimental tasks as Seq2Seq problems that predict the result entities by learning the sequential features of the given sessions with their ranking scores for TopK recommendation.

2) Entity Set Expansion: state-of-the-art methods for entity set expansion issues.

- SetExpan [53] is a corpus-based entity set expansion approach that iteratively extracts the context features using skip-gram method. It provides a rank ensemble mechanism by the entity similarity for entity expansion. We select SetExpan as a baseline because it is a representative entity expansion method that takes context knowledge into account. In the experiments, SetExpan suggests a set of result entities according to the given session of entities. This method uses the context information, yet it ignores the sequential features.
- MCTS-bootstrapping [54] is a bootstrapping entity set expansion framework that combines the Monte Carlo Tree Search (MCTS) algorithm and a deep similarity network. It predicts entities by evaluating delayed

feedbacks for pattern evaluation in a bootstrapping system. In this paper, the MCST-bootstrapping model is selected as a baseline because it uses the MCTS algorithm that is an outstanding decision method for entity suggestions. In the experiments, MCST-bootstrapping computes the pattern embedding between each input entity and the context entity. The model predicts the expanded set entities by computing the pattern similarity. Although this method well models the feedbacks, it rarely applies the structural features of KG.

- RippleNet [55] is an end-to-end algorithm by applying a KG for recommendations in order to enhance the user preference model. The model implements a mechanism that user's potential preferences propagate along graph paths rooted at each user like ripples of water. It also extracts high-order preferences using a memory neural network. This method predicts the possible nodes based on the ripple propagation. We select RippleNet as a baseline because it is a well-known PER method using KG. In the experiments, RippleNet is trained with a session of entities and a context KG. For prediction, the method learns the features of the given entities for the K-hops recommends. Although RippleNet models the structural information of the KG, it does not consider the sequential features of the given session entities.

3) Session-based Recommendation: recommendation models for session-based data.

- Graph Contextualized Self-Attention Network (GC-SAN) [56] is a deep learning-based model for session recommendations. It learns the features of session entities using Graph Neural Network and self-attention mechanism. We select GC-SAN because it learns the hidden relational features of sequence data. In the experiments, GC-SAN builds a directed graph for the given session entities in order to train the model. After training, the model can learn the input session entities and recommend a set of ranked entities. This model focuses on the session features, yet it does not take the knowledge information into consideration.

- BERT4Rec [57] uses BERT to learn the features of behaviour sequences for sequential recommendations. This model extracts the context information from the user behaviour sequences by using the deep bidirectional self-attention mechanism. We select BERT4Rec in this paper because it uses the state-of-the-art pre-trained sequential model to learn session features for recommendations. We train BERT4Rec with entity sequences in the experiments. The trained model can predict a set of result entities for the given sequential entities. BERT4Rec focuses on the sequential features rather than the semantic relations of entities.

D. Metrics

We use the metrics Precision, Recall, F1-measure, and Mean Average Precision.

- Precision: Precision indicates the proportion of the true positive samples in the predicted positive sample set.

$$Precision = \frac{Ture\ Positive}{True\ Positive + False\ Positive} \tag{24}$$

- Recall: Recall indicates the proportion of the true positive samples in the real positive sample set.

$$Recall = \frac{Ture\ Positive}{True\ Positive + False\ Negative} \tag{25}$$

- F1-Measure: F1-measure is defined as a weighted average of the precision and recall. The metric balances both of the precision and recall in one score.

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{26}$$

- Mean Average Precision (MAP): MAP is a widely used metric for evaluating ranked lists in the field of information retrieval. The critical difference between Precision and MAP is that the MAP additionally considers the entire ranking of the list.

$$MAP = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{n}\sum_{i=1}^{k} P@i \times rel_i \tag{27}$$

where rel_i equals 1 if the result at rank i is the correct example and 0 otherwise.

E. Task Settings

We compute the Precision (P@K), Recall (R@K), F1-measure (F1@K), and Mean Average Precision (MAP@K) for the TopK recommendations to evaluate our method.

- Value of *K*: For KdConv-bd, we set K as 5 and 10 because the mean average size of TSG of this dataset is relative large. For CSQA-bd, we set K as 1 and 2 because this database has a small mean average size of TSG that cannot support the bigger values of K.

- Size of CSG and TSG: We set suitable parameter pairs of CSG and TSG for different experimental goals. In general, based on the characteristics of the subgraphs, we set a small size of CSG and a big size of TSG for KdConv-bd, while set a big size of CSG and a small size of TSG for CSQA-bd.

F. Training

For KdConv-bd, we trained our model for 3,000 epochs with ca. 60 hours. For CSQA-bd, we trained our model for 2,500 epochs with ca. 45 hours.

VI. Experimental Results

A. General Comparison

The general performance comparison with the both datasets are presented in TABLE II and TABLE III. In the experiments, we use data with different CSG sizes, while we do not limit the NF length that equals to the corresponding CSG size at most.

Our method mostly has the best performance compared with the baselines across both the datasets. With the comparison of the best performance of the baselines, our method respectively obtains the consistent improvements in the both datasets except with the CSG size of 2. In this special case of the KdConv-bd

dataset, the sequence-related models, including our method, do not work well because the models hardly extract sequential features from the too short sequence data. Oppositely, the graph-based methods can obtain entity features by encoding the context KG, which focus on the low-hop relations.

When the CSG size is greater than 2, our method obviously outperforms the conventional baseline methods that are usually used as basic models of feature learning. For the experiments with KdConv-bd, the precision of our method reaches 0.667 (CSG size = 3) that is quite practical for PERs. Precision has been improved maximum 8.33% by our method (CSG size = 5). Recall of our method achieves 0.684 in the best case (CSG size = 3) and maximum 7.65% improvement (CSG size = 5). Our method has the best MAP of 0.541 (CSG size = 3) and maximum 7.55% improvement (CSG size = 3).

TABLE II PERFORMANCE COMPARISON WITH KDCONV-BASED DATASET

Methods	P@5	P@10	R@5	R@10	MAP@5	MAP@10
CSG size = 2						
PageRank	0.207	0.189	0.278	0.302	0.181	0.168
GAT	0.329	0.272	0.298	0.347	0.364	0.312
Seq2Seq	0.191	0.159	0.232	0.335	0.187	0.16
SetExpan	0.301	0.268	0.288	0.347	0.284	0.247
MCTS-bst.	0.184	0.157	0.285	0.287	0.225	0.181
GC-SAN	**0.368**	**0.362**	**0.332**	**0.375**	**0.389**	**0.361**
BERT4Rec	0.204	0.168	0.246	0.358	0.284	0.224
RippleNet	0.324	0.297	0.302	0.324	0.308	0.274
Ours	0.228	0.186	0.271	0.297	0.263	0.222
Improv.	-38.04%	-48.62%	-18.37%	-20.80%	-32.39%	-38.50%
CSG size = 3						
PageRank	0.286	0.289	0.29	0.313	0.202	0.182
GAT	0.487	0.438	0.357	0.415	0.425	0.401
Seq2Seq	0.308	0.258	0.283	0.384	0.258	0.235
SetExpan	0.621	0.598	0.504	0.638	0.484	0.427
MCTS-bst.	0.614	0.585	0.533	0.644	0.503	0.458
GC-SAN	0.569	0.51	0.524	0.623	0.492	0.424
BERT4Rec	0.435	0.402	0.315	0.404	0.363	0.289
RippleNet	0.588	0.557	0.515	0.614	0.498	0.433
Ours	**0.667**	**0.647**	**0.568**	**0.684**	**0.541**	**0.492**
Improv.	7.41%	8.19%	6.57%	6.21%	7.55%	7.42%
CSG size = 5						
PageRank	0.269	0.261	0.281	0.323	0.195	0.175
GAT	0.42	0.401	0.333	0.389	0.392	0.366
Seq2Seq	0.269	0.224	0.257	0.361	0.223	0.203
SetExpan	0.575	0.532	0.468	0.525	0.402	0.381
MCTS-bst.	0.588	0.541	0.487	0.601	0.477	0.405
GC-SAN	0.514	0.487	0.462	0.502	0.445	0.379
BERT4Rec	0.411	0.388	0.301	0.387	0.314	0.257
RippleNet	0.523	0.503	0.441	0.578	0.463	0.398
Ours	**0.637**	**0.584**	**0.519**	**0.647**	**0.505**	**0.434**
Improv.	8.33%	7.95%	6.57%	7.65%	5.87%	7.16%

For the experiments with CSQA-bd, the precision of our method reaches 0.554 (CSG size = 7) that is also practical for the entity recommendations. The precision has been improved maximum 9.45% by our method (CSG size = 10). The recall of our method achieves 0.381 in the best case (CSG size = 7) and maximum 8.39% improvement (CSG size = 10). Our method has the best MAP of 0.401 (CSG size = 7) and maximum 9.54% improvement (CSG size = 10).

TABLE III PERFORMANCE COMPARISON WITH CSQA-BASED DATASET

Methods	P@1	P@2	R@1	R@2	MAP@1	MAP@2
CSG size = 5						
PageRank	0.232	0.195	0.168	0.235	0.163	0.122
GAT	0.334	0.304	0.267	0.289	0.314	0.227
Seq2Seq	0.201	0.165	0.187	0.215	0.163	0.133
SetExpan	0.468	0.354	0.274	0.334	0.343	0.225
MCTS-bst.	0.474	0.336	0.298	0.338	0.341	0.258
GC-SAN	0.409	0.287	0.286	0.305	0.331	0.203
BERT4Rec	0.352	0.283	0.208	0.248	0.224	0.201
RippleNet	0.421	0.299	0.297	0.311	0.334	0.241
Ours	**0.511**	**0.384**	**0.317**	**0.363**	**0.364**	**0.274**
Improv.	7.81%	8.47%	6.38%	7.40%	6.12%	6.20%
CSG size = 7						

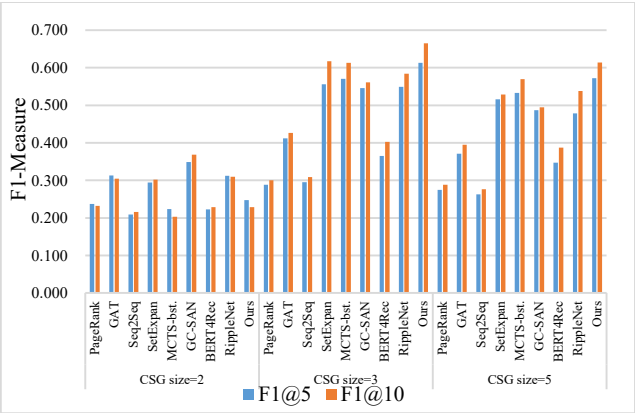
PageRank	0.253	0.221	0.187	0.246	0.171	0.135
GAT	0.387	0.324	0.289	0.301	0.353	0.289
Seq2Seq	0.225	0.208	0.203	0.258	0.186	0.147
SetExpan	0.487	0.325	0.325	0.346	0.348	0.276
MCTS-bst.	0.514	0.411	0.311	0.358	0.375	0.28
GC-SAN	0.469	0.315	0.299	0.341	0.354	0.258
BERT4Rec	0.361	0.318	0.231	0.274	0.242	0.268
RippleNet	0.492	0.363	0.302	0.325	0.362	0.287
Ours	**0.554**	**0.441**	**0.352**	**0.381**	**0.401**	**0.31**
Improv.	7.78%	7.30%	8.31%	6.42%	6.93%	7.27%
CSG size = 10						
PageRank	0.175	0.121	0.156	0.202	0.152	0.114
GAT	0.284	0.268	0.169	0.223	0.289	0.221
Seq2Seq	0.14	0.134	0.114	0.196	0.147	0.129
SetExpan	0.42	0.236	0.241	0.265	0.288	0.187
MCTS-bst.	0.434	0.278	0.268	0.277	0.304	0.237
GC-SAN	0.302	0.254	0.257	0.298	0.256	0.148
BERT4Rec	0.236	0.203	0.152	0.201	0.214	0.174
RippleNet	0.348	0.261	0.244	0.268	0.292	0.196
Ours	**0.475**	**0.302**	**0.284**	**0.323**	**0.333**	**0.257**
Improv.	9.45%	8.63%	5.97%	8.39%	9.54%	8.44%

PageRank takes the worst performance because it computes the explicit relational features without deeply mining the hidden features. *Seq2Seq* learns the sequential features from the session data. *GAT* works on learning the deep structural graph features from KG. The conventional methods only consider one type of features so that their performance are relative low.

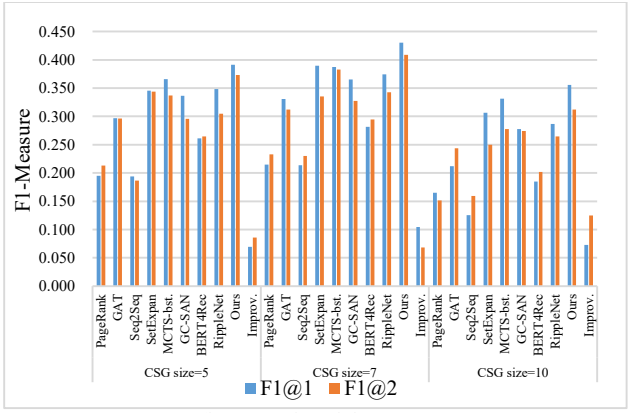
Compared with the recommendation-oriented based baseline methods, we find that our method also obtains significant improvements. The recommendation-oriented baselines achieve better performance than the conventional baselines because these methods not only fusion different types of features but also focus on the optimization of ranking. The session-based recommendation methods, i.e., *GC-SAN* and *BERT4Rec*, efficiently capture the sequential information by encoding the positional information using Graph Convolutional Network or Attention mechanism. The entity set expansion methods usually prefer to encode the features of KG. The experiment results show that the entity set expansion baselines have better precision than the session-based recommendation baselines. It means that the features extracted from KG is more useful than the features learned from the sequence data. Particularly, *MCST-bst.* not only encodes the semantic of KG, also applies the user feedback information. Regarding P@K, *MCST-bst.* can be viewed as a strong baseline. With respect to R@K, *BERT4Rec* has a relative low value because of without considering the graph relations of entities. Other recommendation-oriented baselines have small differences. With respect to MAP@K, *MCST-bst.* and *RippleNet* have relative better values. Therefore, they can be treated as strong baselines for evaluating precision with the order factor. *BERT4Rec* also has the worst value compared with other recommendation-oriented models.

In total, the performance of *MCS-bst.* and *RippleNet* are close to our method. It concludes that the path-based features and user feedback data are critical information for ToKGE.

Fig. 9 presents the results of F1-measure evaluation. Our method outperforms the baseline methods. Our method achieves the satisfied F1 values representing the model has good quality.



(a) KdConv-based dataset



(b) CSQA-based dataset

Fig. 9 F1-measure evaluations

B. *Performance of Our Method*

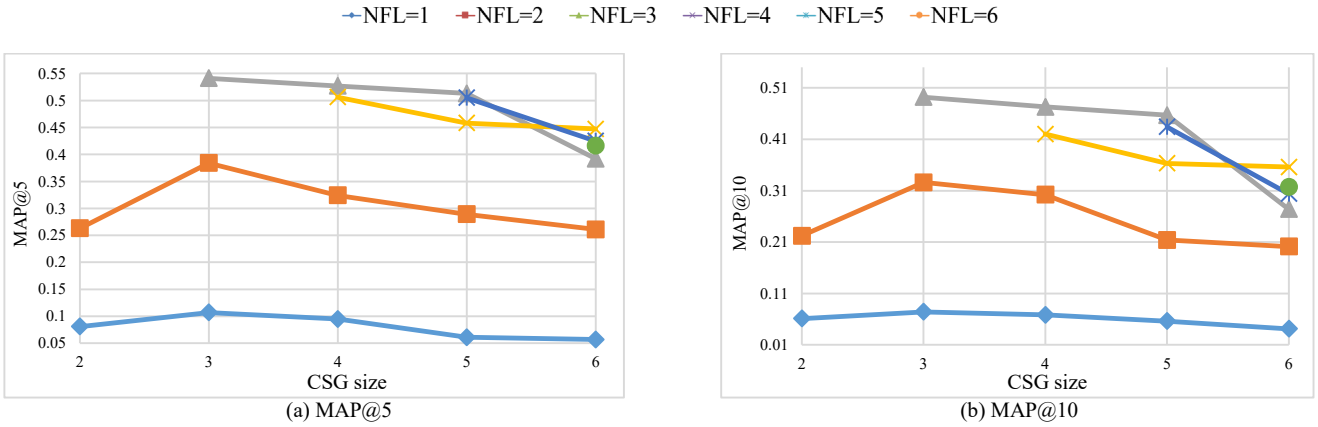
There are more detailed evaluations of our method with MAP by considering the CSG size and NF length in order to analyse the performance patterns. The evaluations start with the CSG size of 2 because a sequence with only one entity is usually not treated as a session. Fig. 10 and Fig. 11 show the results. We find that our method obtains good results when the parameter pairs of the CSG size and NF length are respectively set to (5, 3) and (3, 2) for the experimental datasets.

The experiments show that there exists a set of global optimal settings consisting of an optimal CSG size and an optimal NF length for each sequence data. Taking CSQA-bd as an example, our method obtains the best performance with the optimal CSG size of 7 and the optimal NF length of 4. The difference between the optimal CSG size and the optimal NF length increases along with the growth of the CSG size.

1) Impact of Negative Feedback Length: We find that there is a turning point n of the NF length for a given CSG size. As shown in the Eq. (28), when the CSG size is less than or equal to n, the optimal NF length equals to the CSG size; when the CSG size is greater than n, the optimal NF length is greater than the CSG size.

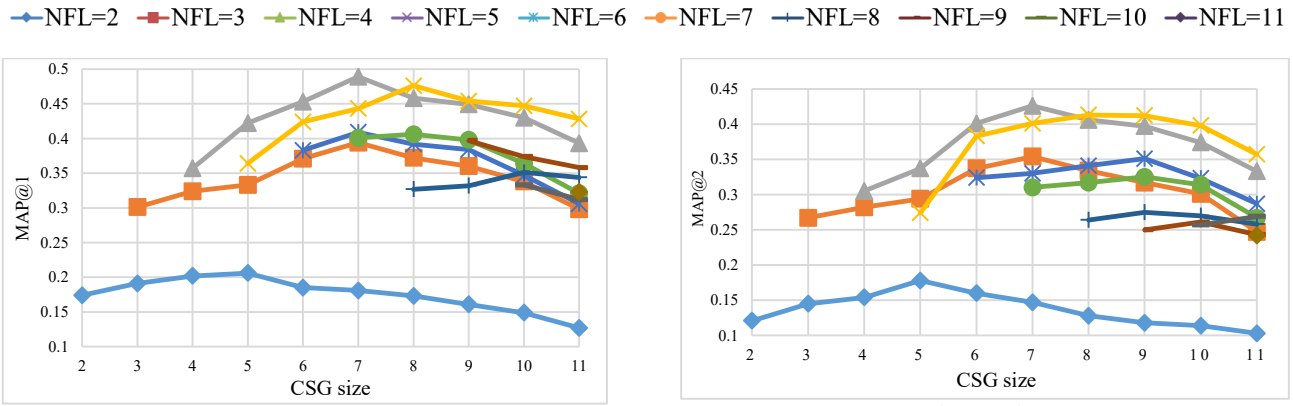
$$\begin{cases} Opt.NF\ length = CSG\ size, & when\ CSG\ size \le n \\ Opt.NF\ length < CSG\ size, & when\ CSG\ size > n \end{cases} \quad (28)$$

For the CSQA-bd, in Fig. 11, we find the turning point $n = 4$. In case of $CSG\ size \le n$, for example, when the CSG size is 3, the optimal NF length is 3; when the CSG size is 4, the optimal NF length is 4. In case of $CSG\ size > n$, for example, when the CSG size is 6, the optiaml NF length is 4. We evaluate impacts of the NF length with the both datasets. In the experiments, the NF length does not exceed the CSG size. From the results shown in Fig. 10 and Fig. 11, we find that the NF length has an optimal value for each dataset. Too short NF length does not really help for modeling user intents while too much negative feedback information will also lead negative influence. In terms of results, the most proper NF length for KdConv-bd is relative bigger than CSQA-bd. We think that it depends on the selected CSG size.



(a) MAP@5 (b) MAP@10

Fig. 10 Model Performance with KdConv-bd.



(a) MAP@1 (b) MAP@2

Fig. 11 Model Performance with CSQA-bd.

2) Impact of Chain-structured Subgraph Size: We study the effect of the CSG size on the performance of our method using MAP. From the *results* shown in Fig. 10 and Fig. 11, we find that our method achieves the best MAP values with the CSG size of 3 for KdConv-bd and the CSG size of 7 for CSQA-based dataset. The performance is increasing before achieving the turning point n, while the performance is decreasing after the turning point. Either too small or too big sizes cannot maximally improve the performance. We conclude that the short session data cannot provide enough information and the longer session contains more noises. Besides, we find that the optimal CSG size is relevant to the mean CSG size of the data. There are less long CSG data in KdConv-bd than in CSQA-bd. Therefore, the optimal CSG size for KdConv-bd is smaller than for CSQA-bd.

C. Ablation Study

We perform an ablation analysis by comparing our method with it's variants. We build three variants respectively by

ignoring negative feedback, intent attention, and knowledge features. The first variant removes NFMN from our method in order to evaluate the impact of the negative feedback. The second variant keeps NFMN but use the fixed weight instead of the intent attention-based weights. The third variant replace the KG embeddings with random embeddings.

We select the proper settings of the CSG size and NF length according to the performance evaluations. We study ablation of the three components with Precision and Recall. TABLE IV shows the results. We find that each of the three variants lead to the performance decrease. It indicates that each type of the ignored information has contribution for the intent modeling.

TABLE IV ABLATION EVALUATIONS

Model		w/o NFMN	w/o Intent Attention	w/o Knowledge Feature	Standard Model
KdConv-based dataset (CSG size=3, NF length=3)					
P@5	value	0.504	0.583	0.611	0.667
	change	-32.34%	-14.41%	-9.17%	
P@10	value	0.491	0.564	0.595	0.647
	change	-31.77%	-14.72%	-8.74%	
R@5	value	0.448	0.508	0.511	0.568
	change	-26.79%	-11.81%	-11.15%	
R@10	value	0.552	0.604	0.608	0.684
	change	-23.91%	-13.25%	-12.50%	
MAP@5	value	0.464	0.491	0.502	0.541
	change	-16.59%	-10.18%	-7.77%	
MAP@10	value	0.43	0.455	0.46	0.492
	change	-14.42%	-8.13%	-6.96%	
CSQA-based dataset(CSG size=7, NF length=7)					
P@1	value	0.454	0.503	0.511	0.554
	change	-22.03%	-10.14%	-8.41%	
P@2	value	0.357	0.392	0.402	0.441
	change	-23.53%	-12.50%	-9.70%	
R@1	value	0.303	0.321	0.318	0.352
	change	-16.17%	-9.66%	-10.69%	
R@2	value	0.321	0.355	0.348	0.381
	change	-18.69%	-7.32%	-9.48%	
MAP@1	value	0.359	0.375	0.374	0.401
	change	-11.70%	-6.93%	-7.22%	
MAP@2	value	0.275	0.283	0.289	0.31
	change	-12.73%	-9.54%	-7.27%	

1) Model without NFMN: The results show that the variant without NFMN largely reduces the performances. It indicates that the negative feedback information provides the critical contributions for encoding the user intent in our method.

2) Model without Intent Attention: The results show that the Intent Attention also reduces the performance. It indicates, although the Intent Attention mechanism is removed, our method provides a lot of valuable features for modeling the user's intents. We find that the effect of Intent Attention is relative smaller than the negative feedback's.

3) Model without Knowledge Features: The results show that the performance obviously reduces in varying degrees with the selected metrics and datasets. It indicates that the knowledge features give significant contributions for our method. The results show that the Recall is more influenced than Precision. We think the reason is that KG can give more features enlarging the search range of the relevant entities.

In general, we find that the perrformance decrease of each variant by MAP is smaller than that by Precison and Recall. It indicates that the three types of modifications do not largely. influece the final rank in our method for TopK recommendations. In sum, the ablation study shows that the features of negative feedback, intent attention, and knowledge graph play importants roles for our method.

D. Case Study

TABLE V shows examples for demonstrating the use of our method. For each example, we sample QA data from KdConv-bd. and transfer to an entity session for Top3 recommendations. The first column of the table shows the request of PER in the formal form of a triple: $[session, last\ topic\ core, ?]$, where the session is described as a list of topics in this example in order to clearly explain the structure of the session; the second columns shows the predictions. The bold entities indicate the correctly predicted results. The results shows that our method can obtain the satisfied performance.

TABLE V EXAMPLES OF OUR METHOD

PER request	Rec. results
[(Chungking Express, Hong Kong Film Awards, Faye Wong, Tony Leung), (Tony Leung, birth date, Infernal Affairs, Happy Together), (Happy Together, Leslie Cheung, singer, Farewell My Concubine)], Farewell My Concubine, ?	**Chen Kaige**, Release time, **Golden Palm Award**
[(Julia Stiles, 10 Things I Hate About You, Dexter, The Bourne Identity3), (The Bourne Identity3, Release time, Matt Damon)], Matt Damon, ?	Harvard University, Cambridge, **Good Will Hunting**
[(Sean Penn, Yale University, Mystic River, Academy Award for Best Actor in a Leading Role, Robin Wright),(Robin Wright, Best Actress of the Golden Globe Award, Forrest Gump), (Forrest Gump, Robert Zemeckis, Academy Award for Best Picture, Tom Hanks)], Tom Hanks, ?	**The Philadelphia Story**, Saving Private Ryan, **Academy Award for Best Actor**

E. Discussion

In sum, our method outperforms the state-of-the-art methods for the PER tasks regarding ToKGE (RQ1). We regard that the larger the CSG size, the better the performance; the larger the TSG size, the better the performance. KdConv-bd has small mean CSG size and quite a few big TSGs, while CSQA-bd is in the opposite situation. Therefore, we regard that the TSG can provide more valuable contributions than the CSG does (RQ2). The experiments show that NFMN is obviously beneficial for the PER tasks of ToKGE. The Intent Attention mechanism and context KG are much valuable as well (RQ3).

We discuss the limitations: (1) our method needs the optimal settings of CSG size and NF length, which strongly depend on the experiences. If CSG size is too small, our method does not work well; however, the performance strongly reduces if CSG size is too big. It indicates that our method needs a better way for encoding sequence data; (2) our method ignores the relations between the negative feedback entities; (3) the number of the candidate entities cannot be large because it will lead high computational complexity when computing intent attention values for each of them; (4) finally, our model has weak parallel computing performance, particularly for the sequence model.

VII. CONCLUSION

Personalized entity recommendations are important for Knowledge Graph exploration to address the information overflow problem. In this paper, we propose a KG-based topic-oriented personalized entity recommendation method integrating positive intents with negative intents. We design a Transformer-based encoder for modelling the adaptive positive intents and a NFMN for encoding the adaptive negative intents. Experimental results confirm the effectiveness of our method. In the future, we will find the good way to determine the optimal parameters of the CSG size and NF length. We will also model the relations among the negative feedback entities. In addition, we will work on improving the computing performance for a large number of candidate entities.

ACKNOWLEDGMENT

This work was supported by National Science and Technology Major Project (2017-I-0006- 0007). We also thank all colleagues and graduate students who helped us for the work.

REFERENCES

[1] Matteo Lissandrini; Davide Mottin; Themis Palpanas; Yannis Velegrakis, Data Exploration Using Example-Based Methods, Morgan & Claypool, 2018.

[2] Matteo Lissandrini, Torben Bach Pedersen, Katja Hose, and Davide Mottin. Knowledge Graph Exploration: Where are We and Where are We going? SIGWEB Newsl., Article 4, 8 pages. Summer 2020.

[3] Hao Zhou, Chujie Zheng, Kaili Huang, Minlie Huang, Xiaoyan Zhu. KdConv: A Chinese Multi-domain Dialogue Dataset Towards Multi-turn Knowledge-driven Conversation. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 7098–7108, 2020.

[4] Serafeim Chatzopoulos, Kostas Patroumpas, Alexandros Zeakis, Thanasis Vergoulis, and Dimitrios Skoutas. SPHINX: a System for Metapath-based Entity Exploration in Heterogeneous Information Networks. Proc. VLDB Endow, pp. 2913–2916, August 2020.

[5] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. Learning Intents behind Interactions with Knowledge Graph for Recommendation. In Proceedings of the Web Conference 2021 (WWW '21). Association for Computing Machinery, New York, NY, USA, pp. 878–887, 2021.

[6] Ruobing Xie, Cheng Ling, Yalong Wang, Rui Wang, Feng Xia, Leyu Lin:Deep Feedback Network for Recommendation. IJCAI, pp. 2519-2525, 2020.

[7] Yuxuan Shi, Gong Cheng, Trung-Kien Tran, Evgeny Kharlamov, and Yulin Shen. Efficient Computation of Semantically Cohesive Subgraphs for Keyword-Based Knowledge Graph Exploration. In Proceedings of the Web Conference 2021 (WWW '21). Association for Computing Machinery, New York, NY, USA, pp. 1410–1421, 2021.

[8] Kuric, E. , JD Fernández, and O. Drozd . Knowledge Graph Exploration: A Usability Evaluation of Query Builders for Laypeople. 2019.

[9] Alan Medlar and Dorota Głowacka. Exploratory Search: User Behaviour and Search Engine Adaptation. by Alan Medlar and Dorota Głowacka with Martin Vesely as Xoordinator. SIGWEB Newsl. Autumn, Article 4 (Autumn 2019), 3 pages, 2020.

[10] L. Zheng, S. Liu, Z. Song and F. Dou, Diversity-Aware Entity Exploration on Knowledge Graph," in IEEE Access, vol. 9, pp. 118782-118793, 2021.

[11] M. Al-Tawil, V. Dimitrova, and D. Thakker, Using Knowledge Anchors to Facilitate User Exploration of Data Graphs, Semantic Web, vol. 11, no. 2, pp. 205–234, 2020.

[12] Jannach, D. , Zanker, M. , Felfernig, A., and Friedrich, G. Recommender Systems: An Introduction. Cambridge University Press, 2011.

[13] Angel Arul Jothi J and Razia Sulthana A. A Review on the Literature of Fashion Recommender System using Deep Learning. International Journal of Performability Engineering,vol. 17, no. 8, pp. 695-702, August 2021.

[14] Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. Interactive Path Reasoning on Graph for Conversational Recommendation. Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. Association for Computing Machinery, New York, NY, USA, pp. 2073–2083, 2020.

[15] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Exploring High-Order User Preference on the Knowledge Graph for Recommender Systems. ACM Trans. Inf. Syst. 37, 3, Article 32, 26 pages, 2019.

[16] Anuja Arora and Anu Taneja. Research Issues, Innovation and Associated Approaches for Recommendation on Social Networks. International Journal of Performability Engineering, vol. 17, no. 12, pp. 1027-1036, December 2021.

[17] Santosh, T. Y. S. S. , Saha, A. , and Ganguly, N. MVL: Multi-View Learning for News Recommendation. SIGIR '20: The 43rd International ACM SIGIR conference on research and development in Information Retrieval. ACM, 2020.

[18] Euna Mehnaz Khan, Md. Saddam Hossain Mukta, Mohammed Eunus Ali, and Jalal Mahmud. 2020. Predicting Users' Movie Preference and Rating Behavior from Personality and Values. ACM Trans. Interact. Intell. Syst. 10, 3, Article 22, 25 pages. September 2020.

[19] Song, C. , Wang, B. , Jiang, Q. , Zhang, Y. , He, R. , and Hou, Y. Social Recommendation with Implicit Social Influence. SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2021.

[20] Lin, Y. , Moosaei, M. , and Yang, H. OutfitNet: Fashion Outfit Recommendation with Attention-Based Multiple Instance Learning. WWW '20. Association for Computing Machinery, New York, NY, USA, pp. 77–87, 2020.

[21] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z. Sheng, Mehmet A. Orgun, and Defu Lian. A Survey on Session-based Recommender Systems. ACM Comput. Surv. 54, 7, Article 154 (September 2022), 38 pages, 2021.

[22] Shoujin Wang, Liang Hu, Yan Wang, and et al. Sequential Recommender Systems: Challenges, Progress and Prospects. In Proceedings of the 28th International Joint Conference on Artificial Intelligence. AAAI Press, pp. 6332–6338, 2019.

[23] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. In ICLR ' 16, 2016.

[24] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19). Association for Computing Machinery, New York, NY, USA, pp. 1441–1450, 2019.

[25] Shu Wu, Yuyuan Tang, Yanqiao Zhu, and et al. Session-based Recommendation with Graph Neural Networks. In Proceedings of the 33rd AAAI Conference on Artificial Intelligence. pp. 346–353, 2019.

[26] Palumbo E., Monti D., Rizzo G., Troncy R., Baralis E. Entity2rec: Property-specific Knowledge Graph Embeddings for Recommender Systems, Expert Systems with Applications, 2020.

[27] Chien-Chun Ni, Kin Sum Liu, and Nicolas Torzec. Layered Graph Embedding for Entity Recommendation using Wikipedia in the Yahoo! Knowledge Graph. In Companion Proceedings of the Web Conference 2020 (WWW '20). Association for Computing Machinery, New York, NY, USA, pp. 811–818, 2020.

[28] Jizhou Huang, Haifeng Wang, Wei Zhang, and Ting Liu. Multi-Task Learning for Entity Recommendation and Document Ranking in Web Search. ACM Trans. Intell. Syst. Technol. 11, 5, Article 54, 24 pages, September 2020.

[29] Yunyi Zhang, Jiaming Shen, Jingbo Shang, Jiawei Han."Empower Entity Set Expansion via Language Model Probing." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics 2020. ACL 2020. pp. 8151–8160. 2020.

[30] Jiaxin Huang, Yiqing Xie, Yu Meng, Jiaming Shen, Yunyi Zhang, and Jiawei Han. Guiding Corpus-based Set Expansion by Auxiliary Sets

Generation and Co-Expansion. Proceedings of The Web Conference 2020. Association for Computing Machinery, New York, NY, USA, pp. 2188–2198, 2020.

[31] Yan, Lingyong, Han, Xianpei, He, Ben and Sun, Le. End-to-End Bootstrapping Neural Network for Entity Set Expansion. Proceedings of the AAAI Conference on Artificial Intelligence. 34. pp. 9402-9409, 2020.

[32] Jifan Yu, Chenyu Wang, Gan Luo, Lei Hou, Juan-Zi Li, Zhiyuan Liu, and Jie Tang. Course concept expansion in moocs with external knowledge and interactive game. In ACL, 2019.

[33] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In Proceedings Of The International Conference on Representation Learning (ICLR 2015), 2015.

[34] Miller, A. H. Fisch, A. Dodge, J. Karimi, A. Bordes, A.;and Weston, J. Key-value Memory Networks for Directly Reading Documents, 2016.

[35] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-RongWen, and Edward Y. Chang. Improving Sequential Recommendation with Knowledge-enhanced Memory Networks. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. ACM, pp. 505–514, 2018.

[36] Shuqi Lu, Zhicheng Dou, Chenyan Xiong, Xiaojie Wang, and Ji-Rong Wen. Knowledge Enhanced Personalized Search. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20). Association for Computing Machinery, New York, NY, USA, pp. 709–718, 2020.

[37] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. Sequential recommendation with user memory networks. In Proceedings of the 11th ACM International Conference on Web Search and Data Mining, 2018.

[38] Bin Wu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. Query suggestion with feedback memory network. In Proceedings of the 2018 World Wide Web Conference. pp. 1563–1571, 2018.

[39] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. Enhancing Re-finding Behavior with External Memories for Personalized Search. In Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM '20). Association for Computing Machinery, New York, NY, USA, pp. 789–797, 2020.

[40] Bahdanau, D. , K. Cho , and Y. Bengio . Neural Machine Translation by Jointly Learning to Align and Translate. Computer Science, 2014.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, pp. 6000–6010, 2017.

[42] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT:Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of NAACL, 2019.

[43] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. Deep Interest Network for Click-Through Rate Prediction. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18). Association for Computing Machinery, New York, NY, USA, pp. 1059–1068, 2018.

[44] Zhou, G. , Mou, N. , Fan, Y. , Pi, Q. and Gai, K. Deep Interest Evolution Network for Click-Through Rate Prediction. Proceedings of the AAAI Conference on Artificial Intelligence 33, pp. 5941-5948, 2019.

[45] Changhua Pei, Yi Zhang, Yongfeng Zhang, Fei Sun, Xiao Lin, Hanxiao Sun, Jian Wu, Peng Jiang, Junfeng Ge, Wenwu Ou, and Dan Pei. Personalized re-ranking for recommendation. In Proceedings of the 13th ACM Conference on Recommender Systems (RecSys '19). Association for Computing Machinery, New York, NY, USA, pp. 3–11, 2019.

[46] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating Embeddings for Modeling Multi-relational Data. In Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13). Curran Associates Inc., Red Hook, NY, USA, pp. 2787–2795, 2013.

[47] Bo, X. , Yong, X. , Liang, J. , Xie, C. , Liang, B. , and Cui, W. , et al. CN-DBpedia: A Never-Ending Chinese Knowledge Extraction System. International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems. Springer, Cham, 2017.

[48] Ownthink. https://www.ownthink.com/ (accessed on 8 March 2022)

[49] Saha, A. , Pahuja, V. , Khapra, M. M. , Sankaranarayanan, K. , and Chandar, S. Complex Sequential Question Answering: Towards Learning to Converse over Linked Question Answer Pairs with a Knowledge Graph, 2018.

[50] Page, Lawrence, Brin, Sergey, and Terry. Page, l. et al. The Pagerank Citation Ranking: Bringing Order to the Web. stanford digital libraries working paper, 1998.

[51] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. In Proc. of ICLR, 2018.

[52] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In NIPS, 2014.

[53] Jiaming Shen, Zeqiu Wu, Dongming Lei, Jingbo Shang, Xiang Ren, and Jiawei Han. SetExpan: Corpus-Based Set Expansion via Context Feature Selection and Rank Ensemble. In ECML PKDD, volume 10534, 288–304, Cham. Springer International Publishing, 2017.

[54] Yan L , Han X , Sun L, et al. Learning to Bootstrap for Entity Set Expansion. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2019.

[55] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18). Association for Computing Machinery, New York, NY, USA, pp. 417–426. 2018.

[56] Xu C, Zhao P , Liu Y , Sheng V. S , and Zhou X. Graph Contextualized Self-Attention Network for Session-based Recommendation. Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19), 2019.

[57] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19). Association for Computing Machinery, New York, NY, USA, pp. 1441–1450, 2019.