Reinforcement Learning Based Path Exploration for Sequential Explainable Recommendation

Yicong Li, Hongxu Chen, Yile Li, Lin Li, Member, IEEE, Philip S. Yu, Fellow, IEEE, and Guandong Xu, Member, IEEE

Abstract—Recent advances in path-based explainable recommendation systems have attracted increasing attention thanks to the rich information from knowledge graphs. Most existing explainable recommendations only utilize static knowledge graphs and ignore the dynamic user-item evolutions, leading to less convincing and inaccurate explanations. Although some works boost the performance and explainability of recommendations through modeling the user’s temporal sequential behavior, most of them either only focus on modeling the user’s sequential interactions within a path or independently and separately of the recommendation mechanism. Moreover, some path-based explainable recommendations use random selection or traditional machine learning methods to decrease the volume of explainable paths, which cannot guarantee high quality of the explainable paths for the recommendation. To deal with the problem, recent path exploration use reinforcement learning to improve diversity and quality. However, unsupervised training leads to low-efficiency path exploration. Therefore, we propose a novel Temporal Meta-path Guided Explainable Recommendation leveraging Reinforcement Learning (TMER-RL), which utilizes supervised reinforcement learning to explore item-item paths between consecutive items with attention mechanisms to sequentially model dynamic user-item evolutions on a dynamic knowledge graph for the explainable recommendation. Extensive evaluations of TMER-RL on two real-world datasets show state-of-the-art performance compared to recent strong baselines.

Index Terms—Reinforcement learning, sequential recommendation, meta-path, explanation.

I. INTRODUCTION

Reasoning with paths over user-item associated Knowledge Graphs (KGs) has been becoming a popular means for explainable recommendations [1], [2], [3]. The path-based recommendation systems have successfully achieved promising recommendation performance, as well as appealing explanations via searching the connectivity information between users and items across KGs. One category of existing works on path-based explainable recommendations seeks auxiliary meta-paths (pre-defined higher-order relational compositions between various types of entities in KGs) as similarity measures and evidence for possible explanations between users and items. However, most existing path-based methods for explainable recommendation simply treat the underlying KGs as static graphs, ignoring the dynamic and evolving nature of user-item interactions in real-world recommendation scenarios. The dynamics and evolutions of users’ interactions with items play a pivotal role in both recommendation precision and explanations for a user’s real intent. Take the scenario in Fig. 1 as an example, Alice purchased a phone case and a phone film after a recent buy of a new phone. If we ignore the sequential information between each purchase (in Fig. 1(a)) and treat the whole information as a static graph (in Fig. 1(b)), the system probably gives an...
explanation of buying the phone case is because a similar customer also bought this phone case by exploring co-purchasing relationships. Whilst the explanation, in this case, might be valid and the underlying reasoning mechanism (collaborative filtering signal) can be used for recommendation, it is still sub-optimal as a more appealing motivation of buying a phone case is due to the recent purchase of a new phone. For this reason, in this example, it is important for a system to be capable of considering temporal and sequential user-item interactions and disentangling the importance of various reasons when generating possible explanations.

Although some prior works have considered some extent of the sequential information for knowledge-aware path-based explainable recommendation problems, they still fail to explicitly model the dynamics of users’ activities. To allow effective reasoning on paths to infer the underlying rationale of a user-item interaction, the method proposed in [2] takes advantage of path connectivity and leverages the sequential dependencies of entities and sophisticated relations of a path connecting a user-item pair. Nevertheless, the methods only consider the sequential information within specific paths and did not consider the importance of the user’s historical sequences in reflecting the user’s dynamic interactions with items. To improve, the KARN model [4] fuses the user’s clicked history sequence and path connectivity between users and items in KGs for recommendation. However, the method models the user’s sequential behaviors and user-item interactions separately and in a coarse-grained manner (treating a user’s click history as a whole), which may restrict the expressiveness of users’ temporal dynamics on recommendation explainability.

In light of this, in this paper, we challenge the problem of exploring users’ temporal sequential dynamics in the context of path-based knowledge-aware recommendation. Different from existing works that either only consider the sequential information within a path or treat the user’s sequential interactions as a whole and separately, we aim to 1) explicitly model and integrate the dynamic user-item interactions over time into the path-based knowledge-aware recommendation, and 2) leverage the captured dynamics of user-item interactions to improve the performance and explainability of the recommendation.

It is worth noting that modeling users’ temporal sequential behavior with path-based knowledge-aware explainable recommendation is a non-trial and challenging task. First, path-based knowledge-aware recommender systems are built upon the well-constructed KGs, and their expected accuracy and explainability are highly related to the underlying KGs and distilled paths. If temporal information between users and items is considered, the original underlying static KGs will be cast into multiple snapshots w.r.t. the timestamps. Compared with the static KGs that consist of all users’ full timelines, each snapshot at a certain timestamp only has partial observations of the global knowledge, which will result in inferior recommendation performance. To deal with the issue, we devise a general framework for temporal knowledge-aware reinforcement path-based explainable recommender systems, namely Temporal Meta-path Guided Explainable Recommendation leveraging Reinforcement Learning (TMER-RL). TMER-RL provides a solution that can explicitly depict users’ sequential behavior while being able to be aware of global knowledge of the entire underlying KGs. Specifically, to model the temporal dynamics between users and items, TMER-RL naturally models the task as a sequential recommendation problem and takes as input users and their corresponding sequential purchase history. To learn the sequential dependencies between consecutive items purchased by a user, TMER-RL novelly explores item-item paths between consecutive items and embeds the paths as context with elaborated designed attention mechanisms to model the dynamics between user and items. Compared to existing path-based explainable recommendation systems that only consider user-item paths as evidence and support for the recommendation decision, TMER-RL contributes another creamy layer the top of existing works, which makes use of the powerful expressiveness of temporal information between users and items.

In addition, when taking paths into consideration for the above purpose, another challenge lies in the existence of a large number of possible paths. It is time-consuming for a model to select several paths that are meaningful, expressive, and have positive impacts on both recommendation performance and explainability. To this end, inspired by prior work [5], [6], our previous work TMER [7] leverages the concept of meta-path [8] and explores diverse meta-path schemas to characterize the context of dynamic interactions between users and items. However, pre-defined meta-paths and random path instances selection from all generated ones involves human and random factors in the following recommendation module. Moreover, the diversity of the explanations is also restricted by the pre-defined meta-path schemas. To solve the above shortcomings, some work adopts reinforcement learning to retrieve diverse paths to improve recommendation. For example, PGPR [1] uses a novel policy-gradient approach to navigate users’ potential interests and form explainable paths, and TPGR [9] regards the candidate nodes as a balanced clustering tree and proposes a tree-structured policy gradient approach to guide the path exploration. However, the above path-finding methods usually suffer from poor convergence because of a lack of supervision and using the brute-force method. The intuitive rationale is that supervision could lead the training process to quickly approach the ground truth and explore a reliable strategy to find paths. Afterwards, ADAC [3] introduces a weakly supervised reinforcement framework to first derive sets of paths according to pre-defined meta-paths and then use reinforcement learning to explore interpretable paths and predict items, whereas it still inevitably adopts the pre-defined meta-path to help fast convergence. To sum up, the above reinforcement learning-based path exploration methods need to not only explore explainable paths, but also predict items for recommendation, so they naturally lack supervision (pre-defined target items) to guide the exploration of the path. However, in our method, the goal of the reinforcement learning method is only to mine diverse paths between the given nodes (users and items), which means our reinforcement path exploration is supervised by the target nodes. Hence, our reinforcement learning path exploration could better converge thanks to the target nodes. We also design a useful score function for paths to assist paths mining.
With respect to the powerful transformer model for sequential modeling, we also elaborate on item-item path attention and user-item path attention units to learn combinational features of multiple paths to further characterize users’ temporal purchasing motivations and their general shopping tastes, respectively. The rationale for developing such path-based attention units is that a user’s motivation toward buying a certain product is complex and consists of multiple factors. For example, when buying a new phone, a customer may consider several factors including a phone’s intrinsic features such as functionality, display, camera, etc., as well as other external factors such as the choices of their close friends, and their previous purchase of certain related electronics using the same operating system. With the help of the designed path-based attention units, the proposed TMER-RL framework is able to learn different weights for various possible paths which will be then used as explanations for recommendations (we show the effectiveness of using the proposed path-based attention units for the explainable recommendation in the experiments by running case studies).

The main contributions of this paper are summarized as follows:

- We analyze the meta-path instances’ context relation learning ability for the recommendation, and differentiate the meta-path instances into user-item meta-path instances and item-item meta-path instances because of their different potential meaning. We introduce, to the best of our knowledge, the first study to model users’ temporal sequential behavior with the item-item path-based knowledge-aware explainable recommendation.

- We propose a novel reinforcement path exploration for diversity path mining. Compared to the pre-defined meta-path derivation, our method could explore more diverse explainable paths, and therefore achieve a better recommendation performance than the random selection from meta-path derived paths.

- We propose Temporal Meta-path Guided Explainable Recommendation leveraging Reinforcement Learning (TMER-RL), which considers users’ dynamic behaviors on top of the global knowledge graph for sequential-aware recommendation and explores both user-item and item-item meta-path paths with well-designed reinforcement framework and attention mechanisms for the explainable recommendation.

- Experiments on two real-world sequential recommendation datasets demonstrate that TMER-RL outperforms several state-of-the-art baselines in recommendation performance and generating path explanations. Especially, we discuss the reinforcement learning-based paths exploration and random selection approaches through the case study.

## II. PROBLEM DEFINITION

In this section, we give some essential definitions and define the problem.

**Definition 1. Heterogeneous Information Network:** In Heterogeneous information network (HIN) $G = (\mathcal{V}, E)$, each edge $e \in E$ represents a particular relation $r \in R$, each entity $v \in \mathcal{V}$ belongs a particular type $\chi \in X$, where $|R| > 1$, $|X| > 1$.

**Definition 2. Meta-path:** A meta-path [8] $P$ in a network from entity $v_0 \rightarrow v_k$, is denoted as $v_0 \xrightarrow{r_0} v_1 \xrightarrow{r_1} v_2 \xrightarrow{r_2} \cdots \xrightarrow{r_{k-1}} v_k$, where composite relation from $v_0$ to $v_k$ is $r = r_0 \circ r_1 \circ r_2 \circ \cdots \circ r_{k-1}$, $\circ$ represents the composition operator on relations.

**Example 1:** As shown in Fig. 2, the product recommendation network $G_P = (\mathcal{V}, E)$ contains 5 types of nodes, including items, users, categories, colors and brands. Edges between users and items denote the buy relation, edges between items and brands denote the is brand of relation, and etc. There are many meta-paths; one of the meta-paths is UIBI, which means user $\rightarrow$ item $\rightarrow$ brand $\rightarrow$ item. In the paper, we explore the meta-path instances instead of meta-paths. A meta-path instance is a specific path, like $u_1 \rightarrow \text{AirPods} \rightarrow \text{Apple} \rightarrow \text{Beats Solo}$ is a meta-path instance of meta-path UIBI.

**Problem 1.** Sequential knowledge-aware explainable recommendation: For a user $u_i \in U$, given the item set $I$, the user transaction sequence $S_u$ and the item associated network $G$, the target of knowledge-aware explainable recommendation is to predict top $k$ items that $u_i$ will interact with, as well as the possible reasoning of recommended items.

## III. METHODOLOGY

In this section, we introduce the proposed model Temporal Meta-path Guided Explainable Recommendation leveraging Reinforcement Learning (TMER-RL). In the remaining of the paper, we use the notation summarized in Table I to refer to the variables and parameters used throughout the paper.

### A. Overview of TMER-RL Architecture

The overall architecture of the proposed TMER-RL model is shown in Fig. 3. It mainly consists of five components. First, to initialize users and items, we use DeepWalk [10] to pre-train user and item entities. Second, we utilize reinforcement learning with a Markov Decision Process (MDP) environment to explore useful and meaningful sequential (temporal) and non-sequential paths to improve the recommendation performance and personalization. In this step, we obtain item-item instance paths between consecutive items using a reinforcement learning framework. In the third step, after embedding instances, we use multi-head attention to learn the weights of instances as the
Fig. 3. It is the architecture of Temporal Meta-path Guided Explainable Recommendation leveraging Reinforcement Learning (TMER-RL). Here shows an example: User Alice bought iPhone 11 pro, Phone case and Phone film in sequential order and the training process for Alice. The index in each network block is the step order of the model.

TABLE I DESCRIPTION OF MAJOR NOTATIONS USED IN THIS PAPER

| Symbol | Description |
|--------|-------------|
| G      | Heterogeneous information network (HIN) |
| R      | Type of edges (relations) |
| T      | Type of nodes (entities) |
| U      | User entity set |
| I      | Item entity set |
| B      | Brand entity set |
| C      | Category entity set |
| A      | Action set |
| p ∈ P  | A meta-path in an HIN |
| u ∈ U  | User u |
| i ∈ I  | Item i |
| W      | Weight matrix |
| h      | Bias vector |
| φ ∈ Φ  | Path instance in a HIN |
| N      | Neighbour function |
| h      | Embedding |
| a      | Attention mechanism parameter |
| r      | User-item rating |

weights of reasoning paths for the specific user. Next, for the item updating, we employ a two-layer attention to make items contain reasoning information. This step also models the users’ sequential purchased information, feeding the previous item’s feature to the next one. Finally, we feed item embeddings, user embeddings, and instances to the recommendation network to make recommendations. The specific steps are elaborated on in the following subsections.

B. Initialize User and Item Representations

We first learn latent representations for involved users and items by treating random truncated walks in a user-item bipartite network as an equivalent of sentences in DeepWalk [10], which optimizes the co-occurrence probability among the entities in a walk by using skip-gram based on word2vec [11]. Other recent advanced graph-based embedding initialization methods can be also applied, like GraphSage [12], GAT [13], and so on. These recent works can usually outperform DeepWalk. However, through an extensive comparison of these methods, DeepWalk is the best choice. The possible reason behind this is that DeepWalk pays more attention to the embedding of nodes in a path, while GCNs, such as GraphSage, GAT, etc., learn the embedding of each entity with the aggregated feature information from its local neighborhood, which means that these approaches pay more attention to local relationships. However, if only considering the recommendation task itself, although the local relationships are significant, the global higher-order relationships learned by walks on a graph also play a notable role. For example, the transitivity of co-purchasing relationships between friends and the substitute relationship of items can be discovered by modeling higher-order relations in a bipartite user-item graph.

C. Incorporating Meta-Path Based Context

In recommendation tasks, external features related to users and items, such as product attributes, user social networks, and user demographic information are usually considered as additional auxiliary information to complement traditional collaborative filtering methods. However, how to utilize the heterogeneous additional information efficiently is an open problem. Some prior works [14], [15], [16] attempted to consider social relations as the user-side information to boost the recommendation performance. To seek help from injecting more complex additional information, recent works [17], [18] introduced meta-paths into recommendation methods to describe relational compositions between various types of entities in heterogeneous information networks. In [17], the authors proposed to diffuse user preferences along different meta-paths in information networks to generate latent features of users and items. Related work [18] first extracts different-aspect features with meta-paths from a HIN, and then fuses aspect-level latent factors to the recommendation systems. However, these methods largely rely on the latent factors obtained from constructed meta-path based similarity matrices, which are too general and only can reflect mutual interaction between different types of entities in a graph but cannot capture the specific information along with particular
path instances. Therefore, inspired by existing work [5], our prior work TMER and the proposed model TMER-RL explore improving both recommendation performance and explainability by modeling more specific meta-path instances.

Different from existing works, we differentiate meta-path instances into two different categories based on the involved entities in a recommendation scenario (i.e., user-item and item-item meta-path instances). Through modeling these path instances, we learn a more detailed meta-path based context to further characterize the motivations, reasons as well as leading factors between each pair of user-item interactions. While previous works such as [5] mainly focus on modeling meta-path instances between a user and an item, this paper highlights the item-item meta-path instances, which we think are beneficial in multiple aspects of sequential explainable recommender systems. First, only considering user-item paths is restrictive for recommendation explainability as user-item paths only represent a user’s general shopping interests. In comparison, item-item paths are more expressive and can reflect diverse reasons by exploring higher-order relations among items, such as complementary products (e.g., phone and a phone case), substitutable items of known items (e.g., iPhone - phone film - Huawei Phone), co-purchased products with other people, etc. In addition, item-item paths sometimes also serve as sequential modeling signals that naturally capture the temporal dependencies between each consecutive item purchased by users, which will be of great impact for sequential explainable recommendations.

Despite the powerful expressiveness of meta-paths in exploring HIN-based knowledge-aware recommendation, it is still challenging mainly because the number of meta-paths is too large to handle (i.e., the amount of edges is cubic to the number of entities). Taking the electric product recommendation scenario for an example, for IU11B1 meta-path schema, if we fix the starting node (iPhone11 pro), there are many instances: iPhone11 pro → Alice → iPad → Apple → AirPods, iPhone11 pro → Amy → Phone case → Miracase → Phone film, and so on. Therefore, it is necessary to explore useful path instances while limiting the total amount to simplify later calculations. The path instances exploration is introduced in the next subsection.

D. Reinforcement Learning for Paths Exploration

Our previous work [7] pre-defines use-item and item-item meta-paths on the recommendation knowledge graph, and randomly selects meta-path instances from all existing ones. The hand-crafted meta-paths not only need human efforts but also are difficult to be determined when dealing with large recommendation knowledge graphs. Therefore, we propose a reinforcement learning module to explore potential useful meta-path instances on the recommendation knowledge graph. The definition of usefulness in our work is to have a contribution to the recommendation module learning. The process of meta-path instances exploration can be defined as a Markov Decision Process (MDP) [19], and we use reinforcement learning to train the exploration policy.

- **State.** It is the status of each step and the state set is defined as $S$. At step $t$, the state $s_t = (e_t, \text{his}_t)$, where $e_t \in (U \cup I \cup B \cup C)$ is the current visiting entity and \text{his}_t = \{e_0, e_1, \ldots, e_t\} is the history visited entities including the current one. Inspired by [1], [20], [21], [22], we add self-loop and inverse edges on knowledge graph $G$ to facilitate graph traversal.

- **Action.** The action set $A_t$ at step $t$ means all candidate entities to go. We define $A_t = N(e_t) \cup \{e_t\}$, which means the candidate entities are the neighbors of the current visiting entity $e_t$ and itself. Moreover, we allocate each action a weight to present the probability of choosing each action. The score function of each action at step $t$ is as follows.

$$s_{e_t} = \text{softmax} \left( \frac{h_{e_t} \cdot h_e^T}{||h_{e_t}|| \cdot ||h_e||} \right),$$

where $h_{e_t}$ and $h_e$ mean the presentation of the current visiting entity and one of the actions, respectively. We believe the cosine function could indicate the similarity of two entities and it is more likely to choose similar entities to form a path on the knowledge graph. However, on the knowledge graph, some entities have dense structures, leading to large action sets. Thus, pruning unimportant candidate entities is necessary to improve efficiency. We rank each action’s score descendingly and obtain the top $k$ as the new action set.

- **Transition.** Given a current state $s_t = (e_t, \text{his}_t)$ and an action $e_t \in A_t$, the transition probability to the next state $s_{t+1} = (e_{t+1}, \text{his}_{t+1})$ is

$$p(s_{t+1}|s_t, e_t) = 1.$$  

- **Reward.** The target of reinforcement learning is to find user-item meta-path instances and item-item meta-path instances, so the target entity is known for each user. Therefore, if we get to the target entity before the predefined maximum step $T$, the reward is 1, otherwise 0, mathematically,

$$R = \begin{cases} 1, & \text{if get to the target entity before step } T, \\ 0, & \text{otherwise} \end{cases}.$$

- **Optimization.** The whole reinforcement learning module maximizes the final reward to learn policy $\pi$ to find useful user-item and item-item meta-path instances, mathematically,

$$J(\theta) = E_{e_0 \in (U \cup I)} (E_\pi (R)).$$

where $e_0$ is the initial entity. We use REINFORCE [23] algorithm to train the objective function.

According to the above MDP framework, we can get candidate user-item and item-item meta-path instances. Taking user-item meta-path instances from $u_n$ to $i_m$ for an example, there should be $p$ candidate paths. We use a score function to calculate each path’s score, mathematically,

$$c_{u_n \rightarrow i_m} = \frac{\sum_{t=1}^{T} s_{e_t}}{T},$$

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Algorithm 1: Reinforcement Learning for Paths Exploration.

**Input:** $S_t = (e_t, h_{n_i})$ is the state at step $t$, $R_t$ is the reward associated with the $t$th transition, $T$ is the maximum step, $C$ is the array storing the candidate paths’ scores, $u_n$ is the start node, $i_m$ is the end node

**Output:** Paths $P$ and Scores $C$

1. for each $t \in [1, T]$ do
2. if $e_t \neq i_m$ then
3. calculate the score $s_{e_t}$ of the candidate nodes according to (1), where $e_t \in A_t$
4. obtain top $k$ candidate nodes as pruned action set $A_t$ according to candidate nodes scores $s$
5. else
6. Break
7. end if
8. end for
9. calculate $R$ according to (3)
10. if $R == 1$ then
11. calculate path’s score $c[u_n \rightarrow i_m]$ according to (5)
12. end if
13. return $P, C$

where $s_{e_t}$ is each step’s action score. The rationale for choosing the average step’s score as the path’s score is that we do not want the length of the path to influence the path’s score. We rank the score of paths descendingly and get the top $q$ candidate paths. The pseudo code of exploring paths from user $u_i$ to the item $i_m$ is shown as Algorithm 1.

### E. Parameterizing Combinational Features of Meta-Paths as Recommendation Context

1) Learning Combinational Path-Based Features With Self-Attention: After obtaining candidate user-item meta-path instances and item-item meta-path instances, we first regard paths as sentences, and nodes as tokens in sentences, using Word2Vec [11] method and Mean$(\cdot)$ operations to learn path embedding. Then, we employ multi-head self-attention units to learn the meta-path based context (the User-Item and Item-Item Path Attention modules shown in Fig. 3). The rationale for deploying such self-attention units is that after reinforcement learning paths exploration, there are still multiple paths between each pair of item-item (or user-item) representing particular distinct reasons (reasoning paths); and we observe that the reasons for buying two consecutive items are not simply unique; rather, the reasons are more complex and likely a mixture of multiple different reasons. For example, the reasons for a customer to buy a phone case right after his/her previous purchase of a new phone are probably a mixture of 1) his/her friends who own a similar phone and bought this particular phone case, 2) the phone case is the most popular match for the purchased new phone, 3) the color of the phone case matches the customer’s preference. The potential reasons can be more than the listed, and again they can be represented by using various meta-paths.

Based on this observation, self-attentive properties of the Transformer model [24], we aim to learn the combinational features from multiple path instances to better characterize the complex reasons between each connected pair of entities in the KG.

$$Attention(Q_\phi, K_\phi, V_\phi) = f \left( \frac{Q_\phi K_\phi^T}{\sqrt{d_k}} \right) V_\phi,$$

$$MultiHead(Q_\phi, K_\phi, V_\phi) = Concat(\text{head}_1, \ldots, \text{head}_m)W^O,$$

where $\text{head}_i$ is $Attention(W_{\text{Q}i}Q_\phi, W_{\text{K}i}K_\phi, W_{\text{V}i}V_\phi)$. $f(\cdot)$ is a softmax function. Query $Q$, key $K$ and value $V$ are self-attention variables associated with path $\phi$, and $W$ is the weight. $d_k$ is the dimensionality (here $d_k = 100$). $Concat(\cdot)$ is the concatenation operation.

2) Modeling Temporal Dependencies With Item-Item Meta-Path Instances: To learn the temporal dynamics of each user, the proposed TMER-RL framework artfully resorts to the above-mentioned item-item meta-path instances together with the well-designed architecture to capture the sequential dependencies between two consecutive items. Compared with most existing works on sequential recommendation [2], [4], [25] that utilize recurrent neural networks to encode the temporal effects between items in a user’s interacted sequence, the proposed TMER-RL bypasses the de-facto default deployment of RNNs or LSTMs that sometimes make the model even heavier. Specifically, the proposed framework novelly models the temporal dependencies between two items by capturing 1) the information passed from the previous item through an item-attention unit, and 2) item-item connectivity through a specific candidate path instance. Notably, the information passed from the previous item is an attentive aggregation of previous item-item connectivity information. For example, in Fig. 3, whether Alice will buy the phone film is modeled by considering 1) the information passed from the phone case (which includes the connectivity between the iphone11 pro and the phone case), and 2) the paths between the phone case to the phone film. As a result, the long-range and short-term “memory” in a sequence can be captured, and the extent of the goodness of the long and short-term memory can be influenced by the length of the modeled sequence in different scenarios with different datasets.

3) Updating Item Representations: After updating representations of user-item meta-path instances and item-item meta-path instances according to a multi-head self-attention mechanism, we employ a different kind of attention mechanism to update item representations. It is obvious that the current item is mostly related to the last one, which means it is better to add the last item’s information to the current one to contain temporal information. Besides, the current item is also related to the instances that arrived at this item. Therefore, we perform a two-layer attention mechanism to update item representations.

Mathematically,

$$h_i^{(1)} = g(W_{i-1}h_{i-1} + W_{\phi_{i-1}}h_{\phi_{i-1-1}} + b_i^{(1)}) \odot h_{i-1},$$

$$h_i^{(2)} = g(W_i h_i + W_{\phi_{i-1}}h_{\phi_{i-1-1}} + b_i^{(2)}) \odot h_i,$$
where $h^{(1)}_i$ and $h^{(2)}_i$ mean the first and second layer output of the item attention module, respectively. $h_{i-1}$ is the last item’s latent representation. $\phi_{i-1 \rightarrow i}$ is the instance from the $(i-1)^{th}$ item to the $i^{th}$ item. $W$ and $b$ with different superscripts denote different variables’ weights and bias. $g(\cdot)$ is ReLU function. However, there is still a problem with the calculation of the first item, because there is no item before it. To solve this problem, we involve the user-item instance from user $u$ to the first item in the update of the first item, as (9) shows. Actually, the instance from $u$ to the first item is really important in the recommendation, since it is the first item that the user has bought and most of the other bought items have a sequential relation with the first item to some extent. That is why we embed it into the first item.

$$h_{i=1} = g(W_i h_i + W_{\phi_{u \rightarrow i}} h_{\phi_{u \rightarrow i}} + b_i) \odot h_{i}, \quad (9)$$

where $\phi_{u \rightarrow i}$ represents the path from user $u$ to the first item.

### F. The Complete Recommendation Model

Finally, we concatenate user, item and instances information (calculated in Section III-E2) to a vector according to (10), and get user-item prediction scores through Multilayer Perceptron (MLP) with explainability instances.

$$h_{u,i} = [h_u; h^{(1)}_i; h^{(2)}_i], \quad (10)$$

where $[;]$ means vector concatenation. Here $h_{u,i}$ denotes the explicit mutual vector of the user, item, and implicit mutual of user-item meta-path instances and item-item meta-path instances. For the first item, the concatenation operation is different because of the dimension problem, and therefore, for the first item related to each user, the vector fed in MLP involves a user-item instance, mathematically,

$$h_{u,i=1} = [h_u; h_{\phi_{u \rightarrow i}}; h_{i=1}]. \quad (11)$$

After that, the final user-item rating $r_{u,i}$ calculates as follows.

$$r_{u,i} = MLP(h_{u,i}), \quad (12)$$

where the MLP contains a two-hidden-layer neural network with ReLU as the activation function and the sigmoid function as the output layer. According to [5], [26], neural network models can learn more abstractive features of data by using a small number of hidden units for higher layers, we employ a tower structure for the MLP component, halving the layer size for each successive higher layer.

We use implicit feedback loss with negative sampling [27], [28] as the whole loss function

$$loss_{u,i} = -E \sum_{j \in P_{neg}} \log (1 - r_{u,j}), \quad (13)$$

where models the negative feedback drawn from the noise distribution $P_{neg}$, which is a uniform distribution following [5]. The whole TMER-RL process pseudo code is shown as Algorithm 2.

### G. Discussion of Explanation for Recommendation With Attention Mechanism

Attention Model [29], [30] was first introduced in machine translation tasks and the attention weights were later widely used in natural language processing tasks as explanations in neural networks [31], [32]. Other than Natural Language Processing (NLP) tasks, the attention mechanism is also a near-ubiquitous method in recommendation tasks used as explanations in some works. [33], [34], [35], [36] However, there are different opinions on whether the attention mechanism could be used as a way to explain data [37], [38], [39].

In our proposed method TMER-RL, the item-item meta-path instances with attention weights learned by the Item-Item Path Attention module in Fig. 3 are used as explanations. To be specific, for all existing item-item meta-path instances, we use reinforcement learning to explore some useful paths, because it is difficult to process all paths whose number is large. After obtaining the candidate item-item meta-path instances, we consider these paths as explanations for purchasing the target item. For example, in Fig. 3, user Alice has bought iPhone 11 Pro, Phone case and Phone film in a sequential order. For paths from Phone case to Phone film, the paths includes Phone case $\rightarrow$ Phone accessories $\rightarrow$ Phone film, Phone case $\rightarrow$
Miracase → Phone film and Phone case → Abby → Phone film. In this situation, the explanations for buying Phone film are that Alice has bought other Phone accessories. Alice has bought other Miracase product, and that Abby has bought the Phone case and the Phone film together. Based on them, we use a self-attention module as Item-Item Path Attention module to learn a distribution of these paths (explanations) to further explain the recommendation. The learned weights are considered as the explainable weighted scores for the candidate item-item instances. Detailed case study is in Sections IV-H1 and IV-H2.

IV. EXPERIMENT

In this section, we present our experimental settings and give analysis of the evaluation results.

A. Experiment Settings

1) Datasets: We use two public data collections to conduct experiments. Amazon datasets [40] and Goodreads dataset [41], [42]. The Amazon dataset contains 29 categories, from which we choose musical instruments dataset, automotive dataset, and toys and games dataset. Each dataset includes brand and category as the metadata. The Goodreads dataset is a public book online rating and review collection. We select mystery thriller crime genre books and choose authors and publishers as the metadata of books. More details are shown in Table II.

We select the latest twelve items for each user and order these items by timestamps, and then we choose the first two items as bridge items, the next four items as training items and the rest as test items. We create the Heterogeneous Information Networks using user, item, category and brand in Amazon datasets, and using user, book, author, and publisher in Goodreads dataset, respectively. Last, user-item meta-path instances and item-item meta-path instances are explored according to Section III-D.

2) Evaluations: We leverage Top K Hit Ratio (HR@K) and Top K Normalized Discounted Cumulative Gain (NDCG@K) as our metrics. For each positive test instance \((u, i^+)\), we mix the ground truth item \(i^+\) with \(N\) random negative items, rank all these \(N+1\) items and compute the HR@K and NDCG@K scores. We set \(N = 500\) and \(K = \{1, 5, 10, 20\}\) for evaluations. For evaluation of the explainability of the recommendation, we use case studies to show the explanations in detail.

3) Baselines:

- **GRU4Rec** [43], [44]: GRU4Rec is a session-based recommendation method using GRU. For each user, we treat the training items as a session.
- **NARRE** [45]: NARRE utilizes neural attention mechanism to build an explainable recommendation system.
- **MRec** [5]: MRec develops a deep neural network with the co-attention mechanism to learn rich meta-path based context information for recommendations.
- **NFM** [46]: NFM effectively combines the linear Factorization Machines (FM) and nonlinear neural networks for prediction under sparse settings.
- **Chorus** [47]: Chorus considers both relations among items and corresponding temporal dynamics to model the recommendation data in a knowledge-aware and time-aware way. The enhanced item representations improve the performance.
- **S^3Rec** [48]: S^3Rec utilizes the intrinsic data correlation to employ self-supervised learning tasks to learn data representations for sequential recommendation enhancement.

B. Implementation Details

For GRU4Rec, we use the ReChorus package [47] to implement the algorithm. For others, we directly run the provided code by each paper. To fairly compare the evaluation results, we modify each baseline’s evaluation code as the same as TMER-RL. Especially, NARRE is a rating prediction model, we turn it into a link prediction model by rating 1 positive item and 500 negative items and ranking them.

C. Parameter Settings

We choose the first and second items for each user as the bridge items, the training or testing process, the last 6 items as the testing positive items, and the remaining 4 are training items. The hyperparameters are carefully tuned to achieve optimal results in all experiments. We implement GRU4Rec, NARRE and NFM based on the details in papers by the PyTorch package. The meta-paths and settings in MRec are the same as the original paper. The meta-paths in FMG are User-Item(UI), User-Item-Brand-Item(UIBI) and User-Item-Category-Item(UICI).

In terms of the TMER compared in the following experiments, which removes the path generation via reinforcement learning module, we use UIBI, UIICI, UIIBICI and UICIBI as user-item meta-paths and ICIBI, ICIBI, ICIICI, IBIBI, IUIUI, ICIUI and IBIUI as item-item meta-paths, where U, I, B and C denote user, item, brand and category, respectively. For our proposed TMER-RL, the learning rate for Amazon Musical Instruments dataset is \(1e-5\), for the Amazon Automotive dataset is \(5e-5\) and for Amazon Toys and Games is \(1e-4\). The parameters for other baselines are searched for their best results. We set the maximum length of path instance exploration as 6 because we set 6 as the max length of meta-path in our previous work [7] and it is convenient to compare the performance.

D. Effectiveness Analysis on Recommendation Results

We compare TMER-RL with 6 other popular and recent baselines, including four sequential recommendations (GRU4Rec, Chorus, S^3Rec and our former proposed TMER), an explainable recommendation (NARRE), a path-based recommendation (MRec) and a factorization machines-based recommendation.
TABLE III

| Datasets                | Metrics      | GRU4Rec | NARRE | MRec | NFM | Chorus | S3Rec | TMER | TMER-RL | Improve |
|-------------------------|--------------|---------|-------|------|-----|--------|-------|------|---------|---------|
| Amazon Musical Instruments | NEG=500     | 0.3579 | 0.6171 | 0.5858 | 0.4457 | 0.1003 | 0.3348 | 0.8620 | 0.8762 | 1.65%   |
|                         | NEG=500     | 0.4833 | 0.8213 | 0.6178 | 0.5291 | 0.3366 | 0.4874 | 0.9559 | 0.9612 | 0.81%   |
|                         | NEG=500     | 0.5244 | 0.8852 | 0.6444 | 0.5505 | 0.6190 | 0.4648 | 0.9642 | 0.9761 | 1.23%   |
|                         | NEG=500     | 0.5377 | 0.9495 | 0.6819 | 0.6193 | 0.6651 | 0.7559 | 0.9736 | 0.9860 | 1.46%   |
|                         | NEG=500     | 0.5910 | 0.7600 | 0.6189 | 0.5208 | 0.4471 | 0.5095 | 0.9389 | 0.9470 | 0.86%   |
| Amazon Automotive       | NEG=500     | 0.4376 | 0.6621 | 0.6899 | 0.6033 | 0.1389 | 0.4467 | 0.7090 | 0.7255 | 2.05%   |
|                         | NEG=500     | 0.7378 | 0.8229 | 0.7295 | 0.6542 | 0.2746 | 0.6483 | 0.9632 | 0.9832 | 2.08%   |
|                         | NEG=500     | 0.7801 | 0.8703 | 0.7646 | 0.6932 | 0.6937 | 0.7722 | 0.9701 | 0.9710 | 0.97%   |
|                         | NEG=500     | 0.5926 | 0.9080 | 0.7876 | 0.7430 | 0.9604 | 0.8511 | 0.9746 | 0.9954 | 1.21%   |
| Goodreads               | NEG=500     | 0.4376 | 0.6521 | 0.6899 | 0.6033 | 0.1389 | 0.4457 | 0.7090 | 0.7255 | 2.05%   |
|                         | NEG=500     | 0.6146 | 0.7450 | 0.7110 | 0.6383 | 0.3587 | 0.5757 | 0.9530 | 0.9734 | 1.93%   |
|                         | NEG=500     | 0.6506 | 0.7604 | 0.7180 | 0.6517 | 0.4386 | 0.6019 | 0.9574 | 0.9761 | 1.95%   |
|                         | NEG=500     | 0.6688 | 0.7699 | 0.7261 | 0.6644 | 0.4746 | 0.6221 | 0.9585 | 0.9773 | 1.96%   |

As shown in Table III, our proposed TMER-RL achieves the best results and in most situations, TMER-RL could always give the correct items among 500 negative items, especially in the Amazon Automotive dataset. The advantages also hold on other datasets. These results demonstrate that our framework can achieve obvious advantages through explicitly modeling each user’s sequential purchased information with the temporal paths, while NARRE and NFM ignore the sequential information for each user. MRec ignores the item-item intrinsic relations and just utilizes user-item interactions to train the recommendation. GRU4Rec utilizes RNNs to model session-based recommendation. Chorus focuses on the way to better learn items’ representations by combining the complement and substitute relations among items on the temporal items. S3Rec uses different ways to pre-train embeddings to improve recommendation. However, the above three sequential recommendations all overlook the relation along paths and only focus on the temporal historical sequence, leading to worse results.

E. Effectiveness Analysis on Modules of TMER-RL

To study the impact of the modules of TMER-RL for the recommendation performance, including the reinforcement learning path generation module, user-item and item-item instances modules, we further compare our model (TMER-RL) with three variants, namely TMER, ¬ UI and ¬ II. TMER removes the reinforcement learning module and generates user-item and item-item instances according to pre-defined meta-paths as mentioned in Section IV-C. ¬ UI means that we do not consider user-item instances and remove the corresponding self-attention module. ¬ II means that we remove the self-attention module for item-item instances. We report the compared results with 100 negative sampling in Table IV.

TMER-RL performs best on all evaluation methods, especially on the Amazon Automotive dataset. We can see that the user-item and item-item instances self-attention modules significantly boost the recommendation method, and adaptively adjust the importance of different instance paths. The self-learned user-item and item-item instance paths contribute to improving the performance of recommendation finally. Especially, using the item-item meta-path instances self-attention module helps TMER-RL improve 5.87% and 6.85% on HR@1 on the Musical Instruments dataset and Automotive dataset, respectively.

TABLE IV

| Dataset               | Metrics     | TMER-RL | TMER | ¬ UI | ¬ II |
|-----------------------|-------------|---------|------|------|------|
| Amazon Musical Instruments | H@10  | 0.9252 | 0.9216 | 0.8797 | 0.8739 |
|                       | H@10       | 0.9942 | 0.9936 | 0.9671 | 0.9471 |
|                       | H@20       | 0.9979 | 0.9965 | 0.9479 | 0.9471 |
|                       | N@01       | 0.9252 | 0.9216 | 0.8797 | 0.8739 |
|                       | N@05       | 0.9741 | 0.9736 | 0.9295 | 0.9247 |
|                       | N@10       | 0.9765 | 0.9761 | 0.9528 | 0.9527 |
|                       | ¬ UI        | 0.9977 | 0.9973 | 0.9310 | 0.9297 |
|                       | ¬ II        | 0.9899 | 0.9896 | 0.9603 | 0.9618 |
| Amazon Automotive     | N@01       | 0.9354 | 0.9343 | 0.8933 | 0.8797 |
|                       | N@05       | 0.9897 | 0.9844 | 0.9419 | 0.9422 |
|                       | N@10       | 0.9908 | 0.9843 | 0.9439 | 0.9445 |
|                       | N@20       | 0.9911 | 0.9863 | 0.9460 | 0.9468 |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Besides, our model can capture sequential information. Compared with randomly generating paths according to pre-defined meta-paths in our training proportion (x-axis in Fig. 5) and compare the NDCG@1 (Fig. 5(a)) and NDCG@20 (Fig. 5(b)) of TMER-RL and TMER.

As we can see, the more training proportion of the full dataset, the better performance of TMER and TMER-RL. This is in line with common sense because a lower training proportion means sparser data, which leads to fewer paths generation and captures less information among users and items, and results in lower performance of recommendation. However, in another perspective, when dealing with the least training proportion data (0.2) in Fig. 5(a) and (b), the gap of TMER-RL and TMER is the most, and the proposed TMER-RL always keeps the better performance. It could get the conclusion that the TMER-RL is more suitable for the sparser dataset, which means it could explore paths in a sparse data environment and make a better recommendation. Besides, the TMER-RL could always outperform TMER on NDCG@1 and NDCG@20 when the training proportion varies, further proving the effectiveness of TMER-RL.

H. Case Study for the Explainability

1) Generating Explainable Paths for Recommendation via Meta-Path: One of this work’s contributions is to give explanations on instances paths while recommending preferable items. This is because our method generates multiple item-item instance paths according to user purchase history, and then it utilizes the attention mechanism to learn the weight of each item-item instance path and aggregate multiple item-item instance paths for each item-item pair. To demonstrate this, we show a random example drawn from TMER on Amazon Musical Instruments dataset.

We randomly select a user whose id is u273 and track his item-item instance paths’ parameters. In the training dataset, u273’s purchase history is i2954, i2280, i4514, and i11158. We show several sampled item-item instance paths with high attention parameters in Fig. 6 and demonstrate our explanations.

- According to Fig. 6, there are three reasons for purchasing i2280. The most probable reason with the highest attention weight is that u273 has bought the last musical instrument category i2954 and i2280 with the attention weight 0.18. For the next item i4514, the reason for purchasing it is that the user u273 has bought i2280 who has the same brand and category with item i4514. There is also only one item-item instance path between some items because the item-brand and item-category data are sparse. Therefore, our method can model the reason through item-item instance paths with different weights.

- Besides, our model can capture sequential information according to user purchase history thanks to item-item instance paths. These item-item instance paths learn the reason path from the current item to the next item. In the whole model, these reason paths will feed to the item attention module. Therefore, our model can recommend learned sequential information.

2) Generating Explainable Paths for Recommendation via Reinforcement Learning: Compared with randomly generating paths according to pre-defined meta-paths in our
It shows $u_{273}$ and related historical purchased items in the upper part of the figure. In the lower part, it displays the reasoning item-item paths along historically purchased items related to $u_{273}$. Green blocks represent the category attributes. Blue blocks represent the brand attributes. Black blocks without physical pictures do not have meta information in the dataset.

We summarize the generated 10 paths to schemes in Fig. 7. The generated paths schemes contain, $\text{IIBICI}$, $\text{IICICI}$, $\text{ICICI}$ and $\text{IICI}$.

Although the number of summarized schemes is less than that of pre-defined meta-paths, the generated paths via reinforcement learning contribute to recommendation more than random generation and selection, which could be proven in the above experiments. To be specific, when randomly generating, the learning paths will be exactly consistent with the pre-defined meta-paths, but it is unsure that the generated paths are useful for the recommendation. However, utilizing reinforcement learning to mine paths could at least guarantee the correlation of nodes on the paths because of the action calculation during exploration. The proposed ranking equation could obtain the $k$ most relevant paths as the explainable paths.

Moreover, the pre-defined meta-paths may not be able to define some more useful schemes. For example, in Fig. 7, there are 4 schemes, but only $\text{ICICI}$ is a pre-defined meta-path in TMER. Furthermore, if the meta-path defines all existing conditions and considers all situations, the search space would be extremely large, leading to high calculating complexity. Therefore, using reinforcement learning to explore schemes is a better choice.

I. User Study for the Explainability

For further studying the explainability of TMER-RL compared with random path generation with pre-defined meta-paths via TMER [7], we conduct a user study to evaluate whether humans prefer the TMER-RL generated explainable paths. Due to limited human efforts, we randomly select 100 pairs of user-item interactions from the Amazon Musical Instruments dataset and let 5 volunteers with some musical instrument knowledge and rich online purchase experience assess the ratio of the intuitive paths generated by two methods and whether the generated paths are intuitive for humans. The volunteers are required to answer the following two questions for each user-item interaction pair.

1. How is the intuitive explainable paths ratio in the generated path set for each method?
2. Which method can generate more intuitive explainable path sets, TMER-RL or TMER?

In the study, for the first question, the volunteers mark the reasonable paths generated by two methods for each user-item pair, and then we calculate the average ratio of the reasonable paths in all generated paths for each method. Further, for each user-item pair, volunteers select which explainable path set generated by two methods is better, and we count the ratio of the better path sets for each method. Statistical results are shown in Table V: 53% of the generated paths through TMER-RL are considered intuitive by assessors, while for TMER, only 42% of paths are categorized as high-quality paths. This validates that given low-quality explainable paths take up a large proportion of all existing paths, reinforcement learning based approach (TMER-RL) still could find more explainable paths, compared to TMER.

**Table V**

|                 | TMER-RL | TMER |
|-----------------|---------|------|
| #Average intuitive path w.r.t each ui pair | 53%     | 42%  |
| #Intuitive path set w.r.t all pairs         | 57%     | 43%  |
with the random selection based TMER. From the whole perspective, compared to the baseline TMER, 57% paths generated by TMER-RL are considered more intuitive and explainable compared to TMER. Specifically, path generation via TMER-RL tends to be accurate. Supposing generating reasons for purchasing Harmonica Holder, TMER-RL generates paths like from category Harmonicas to produce Harmonica and leads to the target product. However, due to the randomness, path generation via pre-defined meta-paths is likely to generate more general information with high-frequency interaction, like the path from the widest category Musical Instruments to the target product, which contains less related information.

V. RELATED WORK

Early recommendation systems mostly rely on Collaborative Filtering (CF) [49], [50], [51], which are based on the idea that users with similar history will be more likely to purchase similar items. However, CF-based recommendations always have sparsity issues and cold-start problems. Therefore, some works utilize side information, like user and item attributes sparsity issues and cold-start problems. Therefore, some works utilize side information, like user and item attributes [52], item contents [53] to solve this issue. Among them, methods based on knowledge graph [1], [2], [4], [54] show great advantages in the recommendation performance and explainability.

Knowledge graph-based recommendations are roughly be categorized into embedding-based approaches and path-based approaches. Prior efforts on embedding-based knowledge graph recommendations [55] always use embeddings of the knowledge graph to model the user-item interactions for recommendations. For example, existing works [56] utilize TranE [57] to embed user-item interactions to integrate knowledge into the recommendation system. Similarly, [58] embeds user and item vectors into the same embedding space for the recommendation. The above approaches model the relations of users and items using knowledge graph embedding methods, which achieves great improvement in model expressiveness. However, these methods are sensitive to the quality of related knowledge graphs.

To this end, meta-paths [8], and the connectivity of different types of nodes are introduced to recommendations, which have the ability to learn the explicit expression along each meta-path schema. In [59], the authors introduce Matrix Factorization (MF) and Factorization Machine (FM) [60] to learn similarities generated by each meta-path for the recommendation. [18] models rich objects and relations in the knowledge graph and extract different aspect-level similarity matrices thanks to meta-paths for the top-N recommendation. Although they achieved appealing performance, the limitation is still obvious that structured meta-path based similarity latent factors can only reflect mutual infectivity along meta-path schemas in a graph but cannot capture the specific information along particular path instances, which limits the explainability of recommendation.

More recently, injecting meta-paths as recommendation context (aggregation of meta-path instances) [5] was introduced for the top-N recommendation. It provides fine-grained explanations based on specific instances. However, it ignores the important sequential dynamics of user-item interactions, which limits the performance of the recommendation performance and interpretability. To consider the sequential information, [2] utilizes LSTM to model the sequential information, but it only considers path-based sequential information between users and items and ignores the importance of the user’s clicked history sequences, which are highly informative to infer user’s preferences. To tackle this issue, [4] attempts to model the sequences of user behaviors and path connectivity between users and items for recommendation. Nevertheless, it only considers user-item paths, which ignores the item-item intrinsic relation information and cannot learn some complex semantic information between items. Moreover, it has to pre-define some meta-paths and randomly sample meta-path instances, which involves too many human efforts and random factors. Based on the above research, we propose a reinforcement learning-based path exploration for recommendation with differentiated user-item and sequential item-item instances to enhance the learning ability for explainability recommendation.

VI. CONCLUSION

We propose TMER-RL, which explicitly models dynamic user-item interactions over time with path-based knowledge-aware explainable capabilities. We explore item-item paths between consecutive items with attention mechanisms for users’ sequential behavior modeling via a reinforcement learning framework. For evaluation, we conduct 6 sets of experiments to prove the effectiveness of TMER-RL, including comparing with 6 state-of-the-art baselines on 2 real-world datasets to show the high performance of recommendation, an ablation test on modules of TMER-RL to analyze the importance of each module, a variable sensitivity test to show the influence of the learning rate varies, the impact of the amounts of training data to prove the ability to process sparse data, case study to explain the way to explain recommendation via generated paths and another case study to show the effectiveness of reinforcement paths generation framework. All the above experiments have proved the explainability, effectiveness, and high performance of the TMER-RL. Future works may include the following directions: 1) explore hyperedge-based explainability in hypergraph structure [61], 2) explore causality learning [62] to discover more appealing paths for explainability, and 3) study and apply RL techniques to improve the explainability [63].

REFERENCES

[1] Y. Xian, Z. Fu, S. Muthukrishnan, G. De Melo, and Y. Zhang, “Reinforcement knowledge graph reasoning for explainable recommendation,” in Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2019, pp. 285–294.
[2] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, “Explainable reasoning over knowledge graphs for recommendation,” in Proc. AAAI Conf. Artif. Intell., 2019, pp. 5329–5336.
[3] K. Zhao et al., “Leveraging demonstrations for reinforcement recommendation reasoning over knowledge graphs,” in Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2020, pp. 239–248.
[4] Q. Zhu, X. Zhou, J. Wu, J. Tan, and L. Guo, “A knowledge-aware attentional reasoning network for recommendation,” in Proc. 34th AAAI Conf. Artif. Intell., 2020, pp. 6999–7006.
[5] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, “Leveraging meta-path based context for top-n recommendation with a neural co-attention model,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2018, pp. 1531–1540.
[6] Y. Zha et al., “Query-based interactive recommendation by meta-path and adapted attention-gru,” in Proc. 28th ACM Int. Conf. Inf. Knowl. Manage., 2019, pp. 2585–2593.

[7] H. Chen, Y. Li, X. Sun, G. Xu, and H. Yin, “Temporal meta-path guided explainable recommendation,” in Proc. 14th ACM Int. Conf. Web Search Data Mining, 2021, pp. 1056–1064.

[8] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, “PathSim: Meta-path based top-k similarity search in heterogeneous information networks,” Proc. VLDB Endowment, vol. 4, no. 11, pp. 992–1003, 2011.

[9] H. Chen et al., “Large-scale interactive recommendation with tree-structured policy gradient,” in Proc. AAAI Conf. Artif. Intell., 2019, pp. 3312–3320.

[10] B. Perozza, R. Al-Rfou, and S. Skiena, “DeepWalk: Online learning of social representations,” in Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2014, pp. 701–710.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” 2013, arXiv:1301.3781.

[12] W. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in Proc. 31st Int. Conf. Neural Inf. Process. Sys., 2017, pp. 1006–1014.

[13] P. Velicković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” 2017, arXiv:1710.10903.

[14] M. Jamali and M. Ester, “A matrix factorization technique with trust propagation for recommendation in social networks,” in Proc. 4th ACM Conf. Recommender Syst., 2010, pp. 135–142.

[15] H. Ma, H. Yang, M. R. Lyu, and I. King, “SoRec: Social recommendation using probabilistic matrix factorization,” in Proc. 17th ACM Conf. Inf. Knowl. Manage., 2008, pp. 931–940.

[16] Q. Yuan, L. Chen, and S. Zhao, “Factorization vs. regularization: Fusing heterogeneous social relationships in top-n recommendation,” in Proc. 5th ACM Conf. Recommender Syst., 2011, pp. 245–252.

[17] X. Yu et al., “Personalized entity recommendation: A heterogeneous information network approach,” in Proc. 7th ACM Int. Conf. Web Search Data Mining, 2014, pp. 283–292.

[18] X. Han, C. Shi, S. Wang, S. Y. Philip, and L. Song, “Aspect-level deep collaborative filtering via heterogeneous information networks,” in Proc. 27th Int. Joint Conf. Artif. Intell., 2018, pp. 3339–3349.

[19] G. Shani, D. Heckerman, R. I. Brafman, and C. Boutilier, “An MDP-based recommender system,” J. Mach. Learn. Res., vol. 6, no. 9, pp. 1265–1295, 2005.

[20] M. Saedi, S. Krieg, C. Zhang, M. Jiang, and N. Chawla, “Heterogeneous relational reasoning in knowledge graphs with reinforcement learning,” 2020, arXiv:2003.06050.

[21] W. Xiong, T. Hoang, and W. Y. Wang, “DeepPath: A reinforcement learning method for knowledge graph reasoning,” in Proc. Conf. Empir. Methods Natural Lang. Process., 2017, pp. 564–573.

[22] R. Das et al., “Go for a walk and arrive at the answer: Reasoning over heterogeneous social relationships in top-n recommendation,” in Proc. 5th ACM Conf. Recommender Syst., 2011, pp. 245–252.

[23] X. Han, C. Shi, S. Wang, S. Y. Philip, and L. Song, “Aspect-level deep collaborative filtering via heterogeneous information networks,” in Proc. 27th Int. Joint Conf. Artif. Intell., 2018, pp. 3339–3349.

[24] R. J. Williams, “Simple statistical gradient-following algorithms for supervised learning,” in Proc. 12th ACM Conf. Rec. Develop. Inf. Retrieval, 2018, pp. 909–912.

[25] M. Wan and J. J. McAuley, “Item recommendation on monotonic behavior chains,” in Proc. 12th ACM Rec. Develop. Inf. Retrieval Syst., 2018, pp. 86–94.

[26] M. Wan, R. Misra, N. Nakashole, and J. J. McAuley, “Fine-grained spoiler detection from large-scale review corpora,” in Proc. 57th Conf. Assoc. Comput. Linguistics, 2019, pp. 2605–2610.

[27] S. Serrano and N. A. Smith, “Is attention interpretable?,” 2019, arXiv:1906.03731.

[28] J. Ni, J. Li, and J. McAuley, “Justifying recommendations using distantly-labeled reviews and fine-grained aspects,” in Proc. Conf. Empir. Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process., 2019, pp. 188–197.

[29] M. Wang and J. J. McAuley, “Item recommendation on monotonic behavior chains,” in Proc. 12th ACM Rec. Develop. Inf. Retrieval Syst., 2018, pp. 86–94.

[30] M. Wan, R. Misra, N. Nakashole, and J. J. McAuley, “Fine-grained spoiler detection from large-scale review corpora,” in Proc. 57th Conf. Assoc. Comput. Linguistics, 2019, pp. 2605–2610.

[31] C. Chen, M. Zhang, Y. Liu, and S. Ma, “Neural attentional rating regression with review-level explanations,” in Proc. World Wide Web Conf., 2018, pp. 1583–1592.

[32] X. He and T.-S. Chua, “Neural factorization machines for sparse predictive analytics,” in Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2017, pp. 355–364.

[33] C. Wang, M. Zhang, W. Ma, Y. Liu, and S. Ma, “Make it a chorus: Knowledge-and-time-aware item modeling for sequential recommendation,” in Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2020, pp. 109–118.

[34] K. Zhou et al., “S3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization,” in Proc. 29th ACM Int. Conf. Inf. Knowl. Manage., 2020, pp. 1893–1902.

[35] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative filtering recommender systems,” in The Adaptive Web, Berlin, Germany: Springer, 2007, pp. 291–324.

[36] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in Proc. 10th Conf. World Wide Web, 2001, pp. 285–295.

[37] Y. Wang, “Survey on deep multi-modal data analytics: Collaboration, rivalry, and fusion,” ACM Trans. Multimedia Comput. Commun. Appl., vol. 17, no. 1s, pp. 1–25, 2021.

[38] S. Gong, “Employing user attribute and item attribute to enhance the collaborative filtering recommendation,” J. Softw., vol. 4, no. 8, pp. 883–890, 2009.

[39] P. Melville, R. J. Mooney, and R. Nagarajan, “Content-boosted collaborative filtering for improved recommendations,” in Proc. 18th Nat. Conf. Artif. Intell., 2002, pp. 187–192.

[40] X. Wang, Y. Xu, X. He, Y. Cao, M. Wang, and T.-S. Chua, “Reinforced negative sampling over knowledge graph for recommendation,” 2020, arXiv:2003.05753.

[41] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, “Collaborative knowledge base embedding for recommender systems,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2016, pp. 353–362.

[42] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, “Improving sequential recommendation with knowledge-enhanced memory networks,” in Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2018, pp. 505–514.

[43] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in Proc. Adv. Neural Inf. Process. Syst., 2013, pp. 2787–2795.
Yicong Li is currently working toward the PhD degree of the Data Science and Machine Intelligence (DSMI) Lab of Advanced Analytics Institute, University of Technology Sydney. His research interests mainly focus on data science, graph neural networks, recommender systems, natural language processing and so on. In particular, her current research is focusing on explainable machine learning, especially the application in the recommendation area. She has published papers in international conferences and journals, such as WSDM, KSEM and IEEE Access.

Hongxu Chen received the PhD degree in computer science from The University of Queensland, in 2020. He is a data scientist, now working as a postdoctoral research fellow with the School of Computer Science, University of Technology Sydney, Australia. His research interests mainly focus on data mining, graph representations, social network analytics and so on. In particular, his research is focusing on learning representations for information networks and applying the learned network representations to solve real-world problems in complex networks such as biology, e-commerce and social networks, financial market and recommendations systems with heterogeneous information sources. He has published many papers in top-tier conferences and journals, such as SIGKDD, ICDE, AAAI, IJCAI, IEEE Transactions on Knowledge and Data Engineering and so on. He also serves as program committee member and reviewers in multiple international conferences, such as KDD, SIGIR, AAAI, and so on, and he also acts as invited reviewer for multiple journals in his research fields, including Transactions on Knowledge and Data Engineering (TKDE), VLDB Journal, IEEE Transactions on Systems and so on.

Yile Li received the bachelor’s degree from Central South University, China, in 2018, and the master’s degree from Tongji University, China, in 2021, in transportation engineering. Her research interests include recommendation-system, graph neural network, reinforcement learning and multi-agent system.

Lin Li (Member, IEEE) received the PhD degree in information science and technology from the University of Tokyo, in 2009. She is a professor with the School of Computer Science and Technology, Wuhan University of Technology, the Distinguished Overseas researcher of Iwate University, Japan, and the senior entrepreneurship Instructor Certified by the Ministry of Human Resources and Social Security. She has long been engaged in research in the fields of information retrieval, recommendation systems and text mining. She has served as a journal editor for Web Intelligence and Human-Centric Intelligent Systems, and a PC or SPC for international conferences such as IJCAI, AAAI, ACM MM, ICME, COLING, EMNLP. She is a reviewer of ACM Transactions on Multimedia Computing, Communications, and Applications, Information Processing and Management, ACM Transactions on Information Systems, WWWJ, Knowledge-Based Systems, and etc.

Guandong Xu (Member, IEEE) is a professor with the School of Computer Science and Data Science Institute, UTS and an award-winning researcher working in the fields of data mining, machine learning, social computing and other associated fields. He is director of the UTS-Providence Smart Future Research Centre, which targets research and innovation in disruptive technology to drive sustainability. He also heads the Data Science and Machine Intelligence Lab, which is dedicated to research excellence and industry innovation across academia and industry. Aligning with the UTS research priority areas in data science and artificial intelligence, he has more than 220 papers published in the fields of data science and data analytics, recommender systems, text mining, predictive analytics, user behavior modeling, and social computing in international journals and conference proceedings in recent years, with increasing citations from academia. He has shown strong academic leadership in various professional activities. He is also the founding editor-in-chief of Human-Centric Intelligent Systems, an assistant editor-in-chief of World Wide Web Journal, as well as the founding steering committee chair of the International Conference of Behavioral and Social Computing Conference.