**Unified Conversational Recommendation Policy Learning via Graph-based Reinforcement Learning**

Abstract:

Current CRS often separately address the decision-making tasks of asking about attributes, recommending items, and timing these actions, resulting in limitations in the scalability, generality, and training stability of CRS.

This paper presents a unified policy learning approach for CRS to handle these three decision-making tasks simultaneously, enhancing the integration of conversation and recommendation components. The authors propose a dynamic weighted graph-based reinforcement learning method to learn policies for action selection at each conversation turn. They also introduce **two action selection strategies** to reduce the candidate action space based on preference and entropy information, improving the sample efficiency.

Introduction

current approaches often separate the decision-making tasks of asking about attributes, recommending items, and timing these actions. This separation can limit the scalability, generality, and training stability of CRS. For instance, these models often require extra efforts to train an offline recommendation model or pretrain the policy network with synthetic dialogue history. Also, the policy learning is often difficult to converge since the conversation and recommendation components are trained separately.

To address these issues, the authors propose a unified approach to policy learning for CRS.

new challenges： such as how to systematically combine conversation and recommendation components for unified policy learning and how to deal with sample efficiency issues given the large action space.

Solution：They suggest leveraging the graph structure of the user-item-attribute network to improve both integration and efficiency.

This approach formulates the three distinct decision-making processes in CRS (when to ask or recommend, what to ask, and which to recommend) as a unified policy learning problem.

UNICORN utilizes a dynamic weighted graph to model the changing relationships between users, items, and attributes during the conversation. It considers a graph-based Markov Decision Process (MDP) environment to handle the decision-making process of recommendation and conversation simultaneously.

**Select item and attributes:**

These strategies ：preference-based item selection and weighted entropy-based attribute selection, consider only potentially important actions, thereby avoiding the need to enumerate the entire candidate item and attribute set.

2 RELATED WORKS

Current studies on Conversational Recommender Systems (CRS) can be categorized into four areas:

1. Exploration-Exploitation Trade-offs for Cold-start Users

2. Question-driven Approaches: These approaches aim to ask users questions to obtain more information about their preferences.

3. Dialogue Understanding and Generation: These studies focus on understanding users' preferences and intentions from their utterances and generating fluent responses for natural and effective dialogue actions.

4. Multi-round Conversational Recommendation: Under this problem setting, the system asks questions about the user’s preferences or makes recommendations multiple times, aiming to achieve engaging and successful recommendations with fewer conversation turns.

Reinforcement Learning (RL) has been widely used in recommender systems due to its ability to consider users' long-term feedback. RL-based recommendation formulates the recommendation procedure as a Markov Decision Process (MDP) of the interactions between the user and a recommendation agent and uses RL algorithms to learn the optimal recommendation strategies.

Graph-based recommendation methods mainly leverage the graph structure to enhance the recommendation performance and model recommendation as a path reasoning problem for building explainable recommender systems. Recent studies have successfully applied graph-based RL methods to various recommender system scenarios. This paper proposes a graph-based RL method for the conversational recommendation scenario based on a dynamic weighted graph.

3 PROBLEM DEFINITION:

The MCR task is formulated as a Markov Decision Process (MDP), where the goal of the CRS is to learn a policy π that maximizes the expected cumulative rewards over the observed MCR episodes. The policy determines the CRS's action a\_t at each timestep, either asking an attribute or recommending items, based on the state representation s\_t learned from the system state and the conversation history, and considering the intermediate reward rt at each step.



4 METHODOLOGY

一張含有 文字, 字型, 螢幕擷取畫面, 圖表 的圖片

自動產生的描述

The proposed method, UNICORN, consists of four main components: Graph-based MDP Environment, Graph-enhanced State Representation Learning, Action Selection Strategy, and Deep Q-Learning Network.

4.1 Graph-based MDP Environment

The MDP environment serves to inform the agent about the current state and possible actions to take, and then rewards the agent based on how well the current policy aligns with the observed user interactions. the MDP environment can be formally defined by a tuple (S, A, T, R), where:

- S denotes the state space.

- A denotes the action space.

- T: S × A → S refers to the state transition function.

- R: S × A → R is the reward function.

4.1.1 State

For the graph-based MDP environment, the state st ∈ S at timestep t contains all the necessary information for conversational recommendation, which includes the previous conversation history and the full graph G containing all users, items, and attributes. For a given user u, two major elements are considered:



 represents the conversation history until timestep t, where:

- P𝑢 denotes the user-preferred attribute.

- Prej and Vrej are the attributes and items rejected by the user, respectively.

2. G𝑢(t) denotes the dynamic subgraph of G for user u at timestep t. The construction of this graph is further explained in Section 4.2.

The initial state s0 is initialized using the user-specified attribute p0, i.e. 

(Note: The graph construction process will be detailed in Section 4.2.)

4.1.2 Action.

把推薦哪一個商品跟ask 哪一個attribute都當做action

According to the state 𝑠𝑡, the agent takes an action 𝑎𝑡 ∈ A, where 𝑎𝑡 can be selected from the candidate item set {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msubsup><mi>V</mi><mrow><mi>c</mi><mi>a</mi><mi>n</mi><mi>d</mi></mrow><mi>t</mi></msubsup></mstyle></math>","origin":"MathType for Microsoft Add-in"} to recommend items or from the candidate attribute set {"mathml":"<math xmlns=\"http://www.w3.org/1998/Math/MathML\" style=\"font-family:stix;font-size:16px;\"><msubsup><mi>P</mi><mrow><mi>c</mi><mi>a</mi><mi>n</mi><mi>d</mi></mrow><mi>t</mi></msubsup></math>","origin":"MathType for Microsoft Add-in"} to ask attributes. Following the path reasoning approach, we have



where is the set of item vertices directly connecting all  (i.e., items satisfying all the preferred attributes), and is the set of attribute vertices directly connecting to one of  (i.e.,attributes belonging to at least one of the candidate items).

4.1.3 Transition.

We consider that the current state 𝑠𝑡 will transition to the next state 𝑠𝑡+1 when the user responds to the action 𝑎𝑡. In specific, if CRS asks an attribute 𝑝𝑡 and the user accepts it, the next state 𝑠𝑡+1 will be updated .

Conversely, if the user rejects the action 𝑎𝑡 ,𝑠𝑡+1 will be updated by 一張含有 字型, 文字, 數字, 白色 的圖片

自動產生的描述or 一張含有 字型, 文字, 白色, 數字 的圖片

自動產生的描述 for 𝑎𝑡 ∈ P or 𝑎𝑡 ∈ V, respectively. As a result, the next state 𝑠𝑡+1 will be 

4.1.4 Reward. our environment contains five kinds of rewards, namely, (1) 𝑟rec\_suc, (2) 𝑟rec\_fail, (3) 𝑟ask\_suc, (4)𝑟ask\_fail, (5)𝑟quit

4.2 Graph-enhanced State Representation

As we formulate conversational recommendation as a unified policy learning problem over a graph-based MDP environment, it is required to encode both the conversational and graph structural information into the latent distributed representations. In order to make use of the interrelationships among users, items, and attributes, we first adopt graph-based pre-training methods to obtained node embeddings for all the nodes in the full graph G.

4.2.1 Dynamic Weighted Graph Construction. As shown in Figure 3, we represent the current state of the graph-based MDP environment as a dynamic weighted graph. Formally, we denote an undirected weighted graph as

G = (N, 𝑨), with the node 𝑛𝑖 ∈ N, the adjacency matrix element 𝑨𝑖,𝑗 denoting the weighted edges

between nodes 𝑛𝑖 and 𝑛𝑗. In our case, given the user 𝑢, we denote the dynamic graph at timestep 𝑡 as ：

一張含有 文字, 字型, 筆跡, 白色 的圖片

自動產生的描述

where {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msubsup><mi>w</mi><mi>v</mi><mi>t</mi></msubsup></mstyle></math>","origin":"MathType for Microsoft Add-in"} is a scalar indicating the recommendation score of the item 𝑣 in the current state. In order to incorporate the user preference as well as the correlation between the asked attributes and the items, such weight {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msubsup><mi>w</mi><mi>v</mi><mi>t</mi></msubsup></mstyle></math>","origin":"MathType for Microsoft Add-in"} is calculated as：  
一張含有 字型, 文字, 筆跡, 行 的圖片

自動產生的描述

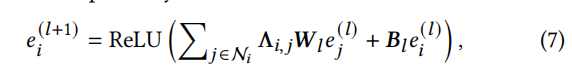
Pu is user preferred attributes & user rejected attritubes?

where 𝜎(·) denotes the sigmoid function, 𝑒𝑢, 𝑒𝑣 , and 𝑒𝑝 are the embeddings of the user, item, and attribute, respectively.

Sigmoid(user-item score + item-attribute score(user preferred attributes) - item-attribute score(user preferred attributes but being rejected) )

4.2.2 Graph-based Representation Learning.

In order to comprehensively take advantage of the correlation information among the involved user, items, and attributes from the connectivity of the graph, we employ a graph convolutional network (GCN) to refine the node representations with structural and relational knowledge. The representations of the node 𝑛𝑖 in the (𝑙 + 1)-th layer can be computed by:



where N𝑖 denotes the neighboring indices of the node 𝑛𝑖, 𝑾𝑙 and 𝑩𝑙 are trainable parameters representing the transformation from neighboring nodes and the node 𝑛𝑖 itself, and 𝚲 is a normalization adjacent matrix as



4.2.3 Sequential Representation Learning. Apart from the interrelationships among the involved user, items, and attributes,the CRS is also expected to model the conversation history in the

current state. Unlike previous studies that adopt heuristic features for conversation history modeling, we employ Transformer encoder for capturing the sequential information of the conversation history as well as attending the important information for deciding the next action. each Transformer layer consists of three components: (i) The layer normalization is defined as LayerNorm(·). (ii) The multi-head attention is defined as MultiHead(𝑸, 𝑲, 𝑽), where 𝑸, 𝑲, 𝑽 are query, key, and value, respectively. (iii) The feed-forward network with ReLU activation

is defined as FFN(·). Take the 𝑙-th layer for example: layer 的构造

一張含有 文字, 字型, 白色, 行 的圖片

自動產生的描述

where 𝑿 ∈ R^𝐿×𝑑 denotes the embeddings, and 𝐿 is the sequence length. In our case, the input sequence {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msup><mi>X</mi><mfenced><mn>0</mn></mfenced></msup></mstyle></math>","origin":"MathType for Microsoft Add-in"}is the accepted attributes {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msubsup><mi>P</mi><mi>u</mi><mi>t</mi></msubsup></mstyle></math>","origin":"MathType for Microsoft Add-in"} in the current conversation history with the learned graph-based representation 

, where 𝐿𝑔 is the number of layers in GCN. After the sequential learning with 𝐿𝑠 Transformer layers, we can aggregate the information learned from both the graph and the conversation history by a mean pooling layer to obtain the state representation of 𝑠𝑡:

一張含有 字型, 文字, 白色, 印刷術 的圖片

自動產生的描述

For simplicity, we denote the learned state representation of 𝑠𝑡 as , where 𝜃𝑆 is the set of all network parameters for state representation learning, including GCN and Transformer layers.

4.3 Action Selection Strategy

we propose two simple strategies to improve the sample efficiency for candidate action selection.

**Preference-based Item Selection**. we select top-𝐾candidate items from  into the candidate action space A𝑡 at each timestep 𝑡, which is ranked by the recommendation score in Eq.(6).

**Weighted Entropy-based Attribute Selection**. Inspired by [14], we adopt weighted entropy as the criteria to prune candidate attributes:

一張含有 文字, 字型, 白色, 行 的圖片

自動產生的描述

where V𝑝 denotes the items that have the attribute 𝑝. Similar to item selection, we also select top-𝐾 candidate attributes from into A𝑡 based on the weighted entropy score 

分子：所有屬於p &在Candidates list的item 他們的score

分母:所有在Candidates list的item 他們的score

衡量屬於p的candidate item的分數佔比

4.4 Deep Q-Learning Network

DQN to conduct the unified conversational recommendation policy learning.

一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述

4.4.1 Dueling Q-Network.

Following the standard assumption that delayed rewards are discounted by a factor of 𝛾 per timestep,

we define the Q-value𝑄(𝑠𝑡, 𝑎𝑡) as the expected reward based on thestate 𝑠𝑡 and the action 𝑎𝑡

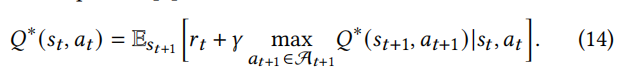
. As shown in the rightmost part of Figure 3, the dueling Q-network employs two deep neural networks to

compute the value function 𝑓𝜃𝑉(·) and advantage function 𝑓𝜃𝐴(·),respectively. Then the Q-function can be calculated by:

一張含有 字型, 文字, 印刷術, 書法 的圖片

自動產生的描述

where 𝑓𝜃𝑉(·) and 𝑓𝜃𝐴(·) are two separate multi-layer perceptions with parameters 𝜃𝑉 and 𝜃𝐴, respectively, and let 𝜃𝑄 = {𝜃𝑉 , 𝜃𝐴}.The optimal Q-function 𝑄∗(𝑠𝑡, 𝑎𝑡), which has the maximum expected reward achievable by the optimal policy 𝜋∗, follows the Bellman equation as:



4.4.2 Double Q-Learning with Prioritized Experience Replay.

During each episode in the MCR process, at each timestep 𝑡,the CRS agent obtains the current state representation  via the graph-enhanced state representational learning described in Section 4.2. Then the agent selects an action 𝑎𝑡 from the candidate action space A𝑡 , which is obtained via the action selection strategies described in Section 4.3. Here we incorporate 𝜖-greedy method to balance the exploration and exploitation in action sampling (i.e., select a greedy action based on the max Q-value with probability 1 − 𝜖, and a random action with probability 𝜖).

Then, the agent will receive the reward 𝑟𝑡 from the user’s feedback. According to the feedback, the current state 𝑠𝑡 transitions to the next state 𝑠𝑡+1, and the candidate action space A𝑡+1 is updated accordingly. The experience

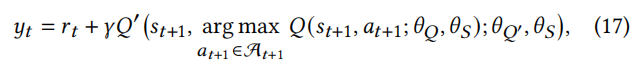
(𝑠𝑡, 𝑎𝑡, 𝑟𝑡, 𝑠𝑡+1, A𝑡+1) is then stored into the replay buffer D. To train DQN, we sample mini-batch of experiences from D, and minimize the following loss function:



where 𝑦𝑡 is the target value based on the currently optimal 𝑄∗.

To alleviate the overestimation bias problem in conventional DQN, we adopt Double Q-learning, which employs a target

network 𝑄′as a periodic copy from the online network. The target value of the online network is then changed to:

  
where 𝜃𝑄′ denotes the parameter of the target network ,argmax for action selection(online network),and the big Q’for target value evaluation .

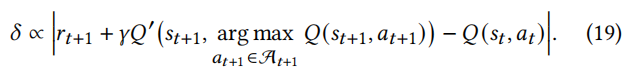
, which is updated by the soft assignment as:

一張含有 字型, 印刷術, 文字, 行 的圖片

自動產生的描述

where 𝜏 is the update frequency.

In addition, the conventional DQN samples uniformly from the replay buffer. In order to sample more frequently those important transitions from which there is much to learn, we employ prioritized replay as a proxy for learning potential, which samples transitions with probability 𝛿 relative to the absolute TD error:



4.4.3 Model Inference.

With the learned UNICORN model, given a user and his/her conversation history, we follow the same process

to obtain the candidate action space and the current state representation, and then decide the next action according to max Q-value in Eq.(13). If the selected action points to an attribute, the system will ask the user’s preference on the attribute. Otherwise, the system will recommend top-𝐾 items with the highest Q-value to the user.

5 EXPERIMENT

Experimental Settings

5.2.1 User Simulator.

we split the ECommerce dataset by 7:1.5:1.5 for training, validation, and testing。𝑟rec\_suc=1, 𝑟rec\_fail=-0.1, 𝑟ask\_suc=0.01, 𝑟ask\_fail=-0.1, 𝑟quit=-0.3.The embedding

size and the hidden size are set to be 64 and 100., the

size of experience replay buffer is 50,000, and the size of mini-batch

is 128. The learning rate and the 𝐿2 norm regularization are set to

be 1e-4 and 1e-6, with Adam optimizer. The discount factor 𝛾 and

the update frequency 𝜏 are set to be 0.999 and 0.01.