RLPer: A Reinforcement Learning Model for Personalized Search

Jing Yao[2], Zhicheng Dou[1], Jun Xu[1], and Ji-Rong Wen[3,4]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]School of Information, Renmin University of China
[3]Beijing Key Laboratory of Big Data Management and Analysis Methods
[4]Key Laboratory of Data Engineering and Knowledge Engineering, MOE
{jing_yao,dou,junxu}@ruc.edu.cn,jirong.wen@gmail.com

ABSTRACT

Personalized search improves generic ranking models by taking user interests into consideration and returning more accurate search results to individual users. In recent years, machine learning and deep learning techniques have been successfully applied in personalized search. Most existing personalization models simply regard the search history as a static set of user behaviours and learn fixed ranking strategies based on the recorded data. Though improvements have been observed, it is obvious that these methods ignore the dynamic nature of the search process: search is a sequence of interactions between the search engine and the user. During the search process, the user interests may dynamically change. It would be more helpful if a personalized search model could track the whole interaction process and update its ranking strategy continuously. In this paper, we propose a reinforcement learning based personalization model, referred to as RLPer, to track the sequential interactions between the users and search engine with a hierarchical Markov Decision Process (MDP). In RLPer, the search engine interacts with the user to update the underlying ranking model continuously with real-time feedback. And we design a feedback-aware personalized ranking component to catch the user's feedback which has impacts on the user interest profile for the next query. Experimental results on the publicly available AOL search log verify that our proposed model can significantly outperform state-of-the-art personalized search models.

CCS CONCEPTS

• **Information systems → Personalization**; • **Computing methodologies → Reinforcement learning**;

KEYWORDS

Personalized Search, Reinforcement Learning, MDP

ACM Reference Format:
Jing Yao[2], Zhicheng Dou[1], Jun Xu[1], and Ji-Rong Wen[3,4]. 2020. RLPer: A Reinforcement Learning Model for Personalized Search. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380294

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '20, April 20–24, 2020, Taipei, Taiwan
© 2020 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380294

1 INTRODUCTION

Search is one of the major approaches for us to obtain information in our daily life. When users enter a query in search engines, they usually have a specific query intent. However, studies have shown that the intent cannot be expressed accurately by the issued keyword query which is usually short and ambiguous [12, 29]. Let us take the query *'MAC'* as an example. A makeup artist may use this query to search for information about the cosmetic brand *'MAC'*, while an IT engineer is likely to seek information about the *'MAC'* computer using the same query. In a current search engine, both users may find unwanted documents which are about 'MAC' but are irrelevant to their real information need. Personalized search is a way to solve this problem by taking user interests into account and returning different results to individual users. So far, there have been many research achievements on this task, including the traditional personalization models [4, 8, 9, 12, 15, 26, 33, 37] and learning based models [13, 27, 32, 34] proposed in recent years.

Typically, a user's personal search process can be regarded as a series of interactions between the user and the search engine: the user inputs a query and the search engine ranks the candidate documents with the ranking model. Then, the user clicks or skips documents and implicates her current interests. During the sequential interactions, the user's interests may dynamically change, and the search engine is expected to generate document lists fitting the user's current interests best. Most existing personalized search models simply regard the search process as a static set of user issued queries as well as the retrieved and clicked documents, and learn fixed ranking strategies with all recorded data. Then, they apply the fixed ranking strategies on new queries without continuous update. They ignore the fact that the user's interests are dynamically changing during the interactions. Though some studies [13, 18] have considered user interests are dynamic, they don't pay attention to the change process and ignore that ranking strategies should also change accordingly. In this paper, we argue that **it would be more helpful for personalization if we model user interests dynamically and update the ranking strategy continuously**.

To tackle this problem, we propose a novel **R**einforcement **L**earning based **Per**sonalized search model **RLPer** in this paper. Within RLPer, we utilize a Markov Decision Process (MDP) [28] to track the interactions between the user and the search engine, and update the personalized ranking model continuously. RLPer provides several advantages for search results personalization. Firstly, it is able to learn the user's dynamic interests better by tracking the search process which contains the variations of user interests. Secondly, RLPer has the ability to dynamically adjust to the user's

current interests as the personalized ranking strategy is updated continuously with the user's feedback in the reinforcement learning framework. Thirdly, compared with existing learning based personalized search models, RLPer can be trained with more training samples (annotated with rewards) through the trial-and-error strategy in reinforcement learning. This is supposed to relieve the problem of limited training samples in personalized search.

E orts have been made on applying reinforcement learning for information retrieval (IR) [36, 39] and recommendation [42 44], but those models are not suitable for personalized search. In this paper, we design our model RLPer fully based on the characteristics of personalized search to better track the interaction process and train the personalized model. We set the search engine as the agent, the user as the environment and a session as an episode to track the dynamic user interests. During a session episode, the search engine interacts with the user in units of query and document list. The user inputs a query and the search engine returns a personalized document list, then the user provides real-time clicks as rewards to the search engine to train the ranking model and the user's interest pro le is updated with these new click behaviors. But it would be better to train our personalized ranking model with the document pairs indicating user preferences, because studies [16] have shown the user's click actions are biased and cannot be used as absolute relevant judgement. To interact with document lists but train the model with document pairs, we formulate the user's search process as a hierarchical Markov Decision Process (MDP). The high level MDP tracks the interaction process by queries and document lists, while all document pairs under each query are sampled as training data to update the ranking model in the low level MDP. In addition, we also design a feedback-aware underlying personalized ranking component which can catch the user's click feedback. Policy gradient algorithm REINFORCE is applied to train RLPer. We experiment with the public AOL search log to compare our proposed model with state-of-the-art personalized search models. The results show that our model can signi cantly improve the e ectiveness of personalized search over existing models.

In summary, our main contributions are three-fold: (1) To the best of our knowledge, it is the rst time that reinforcement learning being applied to personalized search. (2) We carefully adapt reinforcement learning to personalized search and implement a hierarchical MDP model with feedback-aware personalized ranking component, which ts the personalized search scenario better than existing reinforcement learning models for IR. (3) Experimental results on the public AOL logs veri ed that the proposed RLPer model signi cantly improves the quality of personalized search over state-of-the-art models.

The rest of the paper is organized as follows. Related works are reviewed in Section 2. We introduce our model RLPer in Section 3. In Section 4 and Section 5, we discuss the experimental setups and analyze the results. Finally, we conclude the work in Section 6.

2 RELATED WORK

The related works of this paper concern two elds: (1) Search Results Personalization and (2) The Application of Reinforcement Learning.

Search Results Personalization. Personalized search is used to make the search results of ambiguous/broad queries more accurate for each user. Numerous models have been proposed to solve this problem and their basic idea is: First, build user interest pro les by analyzing their query logs. Second, re-rank the candidate documents according to the user pro le and the query. Based on the approaches of building user pro les, we divide existing studies into traditional personalized search models and learning based models.

Traditional personalized search models usually de ne some heuristic rules to analyze the search history and obtain user interests. Focusing on users' re- nding behaviors [30] in search, Dou et al. [12] proposed an e ective method P-Click, which thinks the documents clicked under the same query in the history to be more relevant. Many personalized models [4, 9, 15, 19, 26, 33] adopt a topic model to extract topics from the clicked documents and build user pro les in the topic space. In addition, SLTB [8] manually extracts a lot of features from the query logs to realize search results personalization, including click-based features, query entropy and so on. The user's location and reading level [3, 11] were also applied to achieve personalization. These traditional methods manually extract information from the query logs to build user interest pro les, achieving certain improvements. However, there still exist some drawbacks that they are only able to get user interests covered in these limited features but miss other important information.

As machine learning and deep learning become popular, these drawbacks have been gradually relieved. The learning based models usually learn a representation of the user interest pro le [32] or train a personalized search model [27]. Song et al. [27] and Wang et al. [34] proposed frameworks for adaptation, which adapt the general ranking model to an individual personalized model with a small amount of queries from that user. A hierarchical RNN model (HRNN) [13] was proposed to capture the sequential information in the historical query logs and generate long-term and short-term user pro les related to the current query. PSGAN [18] enhanced the data for personalized model training.

All the aforementioned approaches consider the search process as a static set of user query behaviors and learn xed ranking strategies from the recorded query logs, without continuous change. Di erently, we track the user's entire search process and update the personalized ranking strategy continuously, obtaining a model tting the dynamically changing interests best.

The Application of Reinforcement Learning. Reinforcement learning is usually used to solve problems which can be regarded as a process of sequential decisions or interactions [28]. And it has been widely applied on information retrieval (IR) [22, 36, 39 41] and recommendation [24, 25, 35, 43, 44]. As for the ad-hoc search, Zeng et al. [36] initially proposed a learning to rank (LTR) model based on MDP [28], called MDPRank. This model samples a document to rank at the current position in each step until constructing a ranking list. MDP [39] and multi-armed bandits [22] were utilized to solve the problem of search results diversi cation respectively. In addition, Zeng et al. [41] modeled the multi-page search process as a MDP and took the user's feedback in the former pages to optimize the document list of the next page. However, all these RL based ranking models proposed for ad-hoc search have not taken the user interests, historical and future search behaviors into account. And their datasets always have precise annotations for document relevance, di erent from the noisy data in personalized search.

In recommendation, the process can be naturally regarded as a sequential interactions between the user and the recommend agent. Thus, many studies model the recommendation problem as a MDP and train models in reinforcement learning framework. A MDP-based recommendation system [25] was proposed at an early time to consider the long-term e ect of the current recommended item. Recently, several deep reinforcement learning models [42, 44] were proposed to track the sequential interactions during the recommendation process, trained with the DQN algorithm [28]. Furthermore, a reinforcement learning based interaction interface [14] was designed to facilitate the users to indicate their interests. These tasks share some commonalities with personalized search that all of them are supposed to consider the user interests and history. But the recommend agent provides a single item at a time, di erent from the search engine which needs to sort a document list. And there is no need to think of the relevance with queries in recommendation. In this paper, we fully consider the characteristics of personalized search to design our reinforcement learning model.

3 RLPER - A REINFORCEMENT LEARNING MODEL FOR PERSONALIZED SEARCH

In this paper, we focus on the essence that personal search history is an interaction process between the user and search engine. During this process, the user interests dynamically change. To learn an optimal personalized ranking strategy tting the dynamically changing user interests best, we propose a reinforcement learning based model RLPer to track the user's entire search process and update the personalized ranking strategy continuously.

In this section, we rstly formulate the problem of personalized search as a reinforcement learning process. Then, we introduce our personalization model RLPer and its policy gradient training algorithm in detail. Finally, we describe the online test algorithms.

3.1 Problem Statement

To start with, let us formulate the problem of personalized search as a reinforcement learning process with notations. We use the personalized search engine as the agent and treat the user as the environment. At each time T, the user u which has search history H_T, inputs a query q_T. And the underlying non-personalized search engine returns a candidate document list D_T. Facing the current environment composed of $\{H_T; q_T; D_T\}$, the personalized search engine takes an action a_T. It utilizes its current personalized ranking model M_T to generate a personalized document list D_T^0 with candidate documents in D_T, according to the issued query q_T and the user pro le built on the search history H_T. Then, the user browses the document list D_T^0 to click or skip documents, providing a reward r_T to indicate the quality of the ranking result. The agent then updates the current ranking model M_T to M_{T+1} based on the received reward r_T. And the environment turns into a new state when the user issues a new query q_{T+1}. The new search history includes the last query q_T and search result D_T^0, i.e, $H_{T+1} = H_T + \{q_T; D_T^0\}$. The user interest pro le will also be updated based on the new search history. During this reinforcement learning process, the personalized ranking model is updated based on the user's click feedback continuously until converging to the optimal model.

3.2 RLPer - The proposed Model

The search process described above can be regarded as a sequential decision process during which the personalized search engine decides the order of the documents in the returned document list. Therefore, we mathematically formalize the search process as a MDP, which is always represented as a tuple $(S; A; T; R; \ )$ including the state, action, transition, reward and policy. We design each component of the MDP tuple fully based on the characteristics of personalized search.

(1) In personalized search, we are committed to capturing the user interests throughout the whole search process. Existing reinforcement learning models designed for ad-hoc retrieval [36, 41], which only aim to maximize the immediate return of a single query, are not suitable for this problem. Considering that search sessions are viewed as search activities with independent user intents, we use a session as a MDP episode. We aim to maximize the long-term return of a whole session.

(2) Studies have shown that users' click behaviours are noisy and biased, and clicks cannot be used as absolute relevant judgment [16]. Because of this, the existing pointwise RL approaches [36, 41] may not work well in the personalized search. Instead, we exploit click preferences and adopt pairwise learning to rank algorithm to train the personalized ranking model in RLPer. The agent needs to judge the relative order of a document pair in each step in RL framework.

(3) In the actual interaction process, the search model is expected to return a document list to the user at each time. But it needs to sample all the document pairs under a query as the training data. Therefore, we need to take both the list and document pair levels into consideration in a session episode. To implement interacting with document lists but training the model with document pairs under each query, we design a *hierarchical MDP*. In the hierarchical MDP, we use the high level MDP to track the interaction process in units of queries and document lists, while the low level MDP processes all document pairs under each query. We use T and t as the serial number for the two levels respectively.

In addition, we also design a feedback-aware personalized ranking component called *PHRNN* to capture the user's feedback which will be introduced in Section 3.2.2. The architecture of our proposed RLPer model including the hierarchical MDP structure and the underlying personalized ranking component is shown in Figure 1. The details of each component are introduced as follows.

State S is a set of states describing the environment. As for the high level MDP of interactions, the user inputs a query q_T at each time step T. The search engine (agent) is expected to re-rank the documents based on both the inputted query and the user interests re ected in the search history. Therefore we de ne the state at the step T as $s_T = \{H_T; q_T; D_T\}$. Consistent with the problem statement in Section 3.1, H_T is the user's search history before the query q_T, and D_T is the list of candidate documents for q_T returned by the original non-personalized search engine. In the low level, the search engine needs to train the ranking model with all document pairs under the current query q_T. We use $P_T = \{p_1^T; p_2^T; \ldots\}$ to represent the set of document pairs comprised of all documents in D_T, and the model needs to determine the relative order of a document pair in each step t. So we have $s_t^T = \{H_T; q_T; D_T; p_t^T\}$.

Figure 1: Illustration of the RLPer model. The hierarchical MDP is on the top whose high level tracks the sequential interactions in units of query and document list, and low level trains the ranking model with document pairs under each query. The proposed ranking component PHRNN used to compute the personalized scores for documents is at the bottom. Each query in the search history is represented by a series of document pairs.

Action A is a set of actions for the agent to select, which depends on the current state s_t and is also denoted as $A(s_t)$. In the high level, the search engine is required to return a personalized document list D_T^0 to the user at each step T, according to the personalized score of the documents calculated by the current ranking component M_T. In each step t of the low level MDP, we consider the agent to compare the relevance of the two documents in the document pair $p_t^T = (d_i, d_j)$. Thus, the action set $A(s_t^T)$ can be defined as all possible relationships of the two documents $\{(d_i > d_j), (d_i = d_j), (d_i < d_j)\}$, and the search engine samples an action a_t^T to determine their relative relationship.

Transition $T(S, A)$ is a function $T : S \times A \to S$ which maps the current state to the next state after taking an action. As for the high level MDP of the interaction, the search engine takes an action to return a personalized document list D_T^0 to the user, and the user browses the ranking list and clicks to provide real-time feedback. Then, this query and document list with real-time clicks are added to the user's search history for building new user profile. With the clicked document list, we create a set of document pairs P_T, and the agent takes action a_t^T to judge the relative relationship of the two documents in the pair p_t^T step by step in the low level MDP. All the document pairs in P_T, the actions and the corresponding rewards are collected to update the personalized ranking model from M_T to M_{T+1}. When the user inputs a new query q_{T+1}, the high level MDP turns into the next state with new historical data. Consequently, the transition function T for the hierarchical MDP works as the following equations:

$$s_{t+1}^T = T(s_t^T, a_t^T) = f(h_T, q_T, D_T, p_{t+1}^T), \tag{1}$$

$$s_{T+1} = T(s_T, a_T) = f(h_T + f(q_T, D_T^0), q_{T+1}, D_{T+1}). \tag{2}$$

Reward $R(S, A)$ provides supervision signals for the model training in reinforcement learning, used to measure the influence of actions. Due to we focus on using document pairs as the training data, we refer to the state-of-the-art pairwise LTR algorithm LambdaRank [6] to design our rewards. In LambdaRank, there is a matrix Δ where each element $\Delta_{i,j}$ means the difference between the metric values before and after exchanging the documents d_i and d_j in the ranking list. This matrix reflects the relative relationship of the documents. Different from those supervised learning models which calculate the matrix Δ on the document list recorded in the query log, we calculate it based on the currently returned personalized document list D_T^0 in the interaction. Such real-time feedback reflects the user's current interests which can help RLPer train the personalized ranking model better. We give a positive Δ to the document pairs that are judged correctly by the model and a negative Δ to wrong pairs.

Policy $\pi(a|s) : A \times S \to [0, 1]$ is a probabilistic distribution over the action set calculated with the current state, used as the policy to direct behaviors. Actions of the high level MDP directly depends on the personalized scores. We only need to compute the policy for actions in the low level MDP. From the descriptions above, we know that any action $a \in A(s_t^T)$ corresponds to a possible relative order