PISDR: Page and Item Sequential Decision for Re-ranking Based on Offline Reinforcement Learning

Editors: XXX

Abstract

Re-ranking is the last part of a multi-stage recommendation system, involving the reordering of lists based on historical user behavior to better align with user preferences. Offline Reinforcement Learning (RL) has been employed in both the prediction and ranking phases of recommendation systems to align with long-term objectives, surpassing the efficacy of supervised learning. However, extrapolation error is a common problem in offline RL, due to the biased distribution of features, which can lead to the reduction of recommendation accuracy. Consider that as users browse an e-commerce app, their preferences are influenced by previously recommended items or pages, therefore the history can be used to correct the bias of offline RL. This paper uses offline RL to model re-ranking and presents a re-ranking algorithm named Page and Item Sequential Decision for Re-ranking (PISDR) to improve accuracy by correcting bias at two levels (pages and items). PISDR employs sequential RL, leveraging a session-level data structure that encapsulates global information at the page level and item-level interrelationships. Additionally, PISDR utilizes a multi-tower critic network to assess various feedback metrics, including click-through rate, conversion rate, etc. which can raise actor network from the long-term reward. Experimental results validate the effectiveness of PISDR in significantly enhancing of Area Under Curve (AUC), Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG) about 1.4% in generated re-ranking sequences when compared to current state-of-the-art re-ranking algorithms. Finally, as a consequence, our method achieves a significant improvement (2.59%) in terms of Click-Through Rate (CTR) over the industrial-level ranking model in online A/B tests.

Keywords: Recommendation System; Re-ranking; Offline Reinforcement Learning.

1. Introduction

The Re-ranking is the final stage in a multi-stage recommendation system, where the initial list of the ranking stage is input and a reordered list is output that takes into account the listwise context of e-commerce applications. The primary objective of re-ranking is to elevate the user experience by adeptly recommending items that resonate with individual user preferences while simultaneously optimizing for more strategic, long-term objectives. In the domain of re-ranking, Reinforcement Learning (RL) has emerged as an innovative methodology, distinguished by its ability to optimize for cumulative rewards over time Wei et al. (2022); Wang et al. (2022). Some short-video recommendations in the KuaiShou app or food recommendations in the Meituan app have begun to use RL based methods and achieved a noticeable improvement in terms of total viewing time compared to supervised learning methods Afsar et al. (2022).

RL-based recommendation systems exhibit the capability to manage sequential and dynamic user interactions within the recommendation system, while also accommodating

© 2024 .

the consideration of long-term user interests Afsar et al. (2022); Lin et al. (2023). For instance, Value-based RL methods estimate the user's expectations regarding recommended items by considering user features within the candidate item set. These methods select the item with the highest expected reward for recommendation Zou et al. (2020); Wei et al. (2023); Timmaraju et al. (2023). In contrast, policy-based RL methods directly learn the optimal policy for maximizing the user's expected reward **?**Gao et al. (2023a). As a combination of value-base and policy-base, the Actor-Critic approach involves training two networks: the actor, a policy-based network, responsible for generating recommended items, and the critic, a value-based network, which assesses the actions taken by the actor in response to the user's current state Wang and Wang (2021); Liu et al. (2020a); Cai et al. (2023); Liu et al. (2023). Currently, RL is primarily applied to single-item recommendation or click-through rate prediction tasks, as seen in KuaiShou and Meituan app. This paper stands apart from single-item recommendation scenarios and delves into the optimization of re-ranking list generation within the e-commerce app directly through RL.

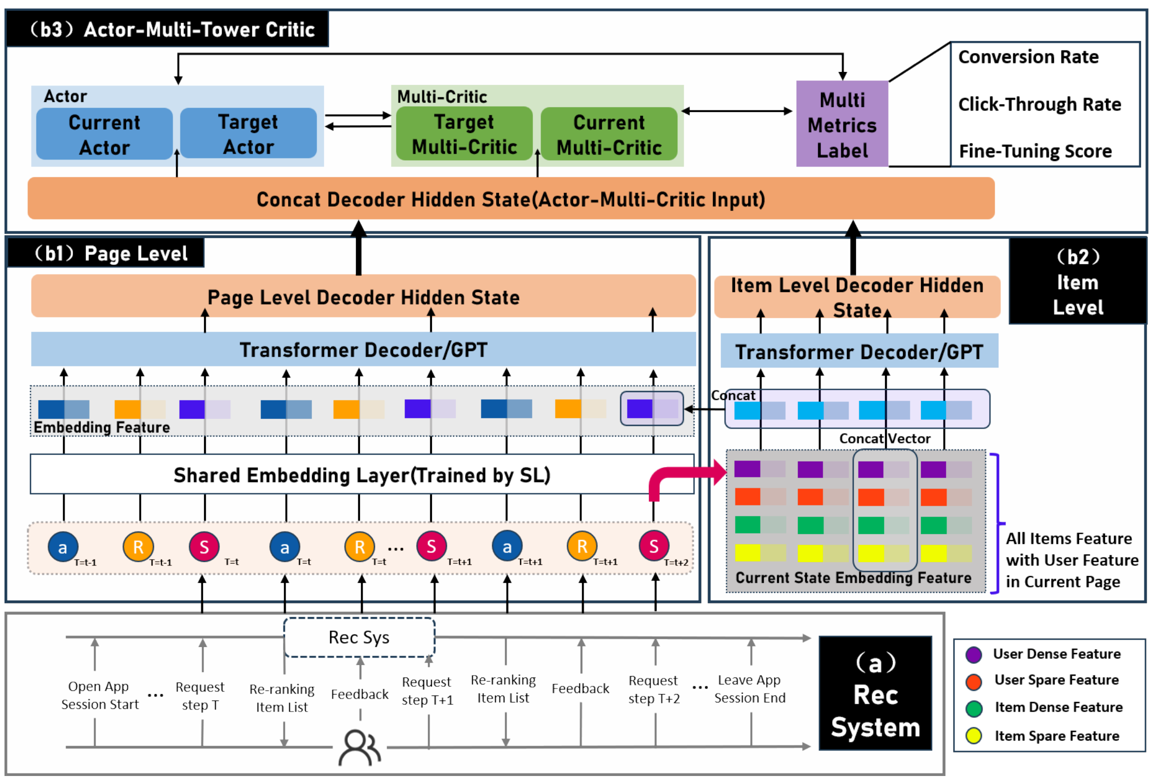

Figure 1: The overall framework, where module (a) is the interaction process between the recommendation system and the use, (b1-b3) is the main architecture of PISDR, where sequential decisions are made at the page-level (b1) and item-level (b2) respectively.

Nonetheless, the direct application of current RL algorithms to generate re-ranking sequences within recommendation systems faces several significant challenges. **1) Extrapolation Error:** The inaccuracies in Offline RL that arise when predicting user preferences outside the training data distribution. **2) Reward Function:** The re-ranking task primarily relies on an accuracy-based evaluation metric, focused on short-term gains, lacking the capability to assess long-term benefits effectively. To mitigate the issue of extrapolation error inherent in RL re-ranking algorithms, we propose a novel PISDR method as shown in Figure (1). To alleviate the extrapolation error in Offline RL, PISDR uses sequential decisions to reduce current decision bias through historical behavior. To match user behavior more closely, PISDR makes decisions through two levels of history, the page(global) and the item(local). Specifically, we design long-term rewards through multiple metrics for the re-ranking problem and construct a multi-tower critic for multiple metrics. Our main contributions are as follows:

1) In this paper, we present a novel model PISDR that focuses on the recommendation system re-ranking at page (global) and item (local) levels. Digging into user interests by processing user trajectories at the page level and item level separately.

2) We propose an offline RL algorithm for re-ranking using a decision transformer approach. This algorithm considers rewards, actions, and states at both the page and item levels. Different from model-based learning, combined with Decision-Transformer, PISDR directly utilizes offline datasets to reduce the extrapolation error through the interaction trajectory.

3) To improve the re-ranking effectiveness at a long-term gain, this paper adopts multiple metrics to design the reward function. We employ a multi-tower critic to evaluate the expected reward of multiple metrics. The code is open source at https://anonymous.4open.science/r/PISDR-832B.

2. RELATED WORKS

In this section, we will introduce related works on task re-ranking for recommendation systems, as well as work related to offline RL in other domains.

2.1. Re-ranking in Recommendation

There are two main branches of neural network based re-ranking models. The first one is supervised learning (SL) in which low-dimensional dense features of users and items, cross-item interactions Bello et al. (2018); Li et al. (2022); Liu et al. (2020b) are extracted by the Recurrent Neural Networks (RNN), self-attention, or Graph Neural Networks (GNN) to generate scores for re-ranking. Typical SL-based re-ranking methods can be classified into two types. The first type is the step-greedy re-ranking strategy, where the items assigned to each position are determined sequentially through serialization. This approach often employs pointer-network Bello et al. (2018) or graph recurrent neural network (GRN) Zhuang et al. (2018) models to generate item IDs for each position one by one. The second type of approach involves re-ranking the item list based on context-wise information. In this method, the relationships of item-item and user-item are utilized to predict the click-through rate (CTR) of each item. PRM Pei et al. (2019) and DLCM Ai et al. (2018) take the initial ranking list as input and use RNN or self-attention mechanisms to model the relationship between contextual information, clicked labels, and predictions. Some of the SL-based models treat the user's behavior history as extra features through a low dimensional embedding layerFeng et al. (2021b). MIR incorporates users' historical behaviors to model set-to-list interactions while considering personalized long-short term interests, aiming to better understand the user's preferences over time Xi et al. (2022). PIER Shi et al. (2023) employs an end-to-end re-ranking framework based on full permutation.

The other method is based on the evaluator-generator to handle some unobserved counterfactual rankings. The generator outputs feasible permutations, while the evaluator scores the results of each permutation. This method is a combination of SL and adversarial learning and can be optimized for a wide range of metrics by using the evaluator Chen et al. (2023); Du et al. (2018). However, both evaluator-generator and SL-based re-ranking approaches are limited to single recommendation tasks and cannot directly optimize long-term metrics like retention and total number of clicks.

2.2. Offline Reinforcement Learning

Online RL requires real-time interaction with users during training, consuming online computational resources and resulting in degraded user experience. As an alternative, offline RL can be used, where log feedback is utilized for training without consuming online resources. However, offline RL is susceptible to challenges such as unobserved logging strategies, as well as issues related to extrapolation errors. Wang and Wang (2021); Wang et al. (2023) proposed a stochastic Actor-Critic method based on a probabilistic formulation and adopted some regularization methods to alleviate the extrapolation error in the recommendation system.

Typical offline RL can be categorized into two types: model-free and model-based. To address the problem of extrapolation errors in offline RL, model-free approaches such as BCQ Fujimoto et al. (2019) use a generative model to limit the probability of state-action pairs used by the policy and avoid using low-frequency data, and CQL Kumar et al. (2020) uses a conservative strategy that includes a penalty for overestimating the Q-value of state-action pairs that do not appear in the offline data. On the other hand, model-based approaches like MOPO Yu et al. (2020) train a pessimistic dynamics model to train a conservative critic. COMBO Yu et al. (2021) learns value functions based on offline datasets and model-generated data, and suppresses value functions for model-generated Out-Of-Distribution (OOD) data.

The extrapolation error problem of offline RL easily leads to the Matthew effect in recommendation systems, to alleviate the Matthew effect, Gao et al. proposed a model-based offline RL method DORL Gao et al. (2023a), which alleviates the Matthew effect in the form of a reward function penalizing items. While there have been various efforts in offline RL in recommendation systems Chen et al. (2019); Gao et al. (2023b); Jeunen and Goethals (2021), only a few works do RL at the re-ranking stage, the evaluator-generator approach is used to do re-ranking in CMR, but still at the level of individual items rather than at the level of contextual information item pages.

3. OFFLINE RL IN RE-RANKING

3.1. Markov Decision Process in Re-ranking

We first define the Markov Decision Process (MDP) for the re-ranking stage of the recommendation system at the session level, as shown in Fig. 1(a), the recommendation system acts as the agent and the user acts as the environment, when the user opens the app, it is the beginning of a session. At each user request t, the recommendation system takes the action of reordering the list of items according to the result of the re-ranking phase to display to the user. After that, the user's feedback are used as reward metrics. The MDP in a recommendation system can be represented in the form of a quintuple $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$, where the specific meanings are as follows:

- **State $\mathcal{S}$:** The user's current state can be represented by incorporating various factors $s_t = \{U_t, I_t, H_t\}$, including the user's dense and spare profile U_t, the dense and spare characteristics of the items in the initial list I_t, and the attributes of the items that the user has previously clicked on H_t.
- **Action $\mathcal{A}$:** Map the input to a list of re-ranking scores $a_t = \phi(\mathcal{V}_t) \in \mathbb{R}^N$ based on the initialized list of N items in the input $\mathcal{V}_t = \{1, 2, ..., N\}$.
- **Transition Probability $\mathcal{P}$:** $P : S \times A \to \Delta(S)$ is the state transition probability, after the recommendation system agent receives the user's feedback such as click, the state transits from s to s' according to the probability $p(s'|s, a)$.
- **Reward Function $\mathcal{R}$:** $R : S \times A \to \mathbb{R}^m$ is the vector-valued reward function which represents m different reward $r(s_t, a_t) = (r_1(s_t, a_t), ..., r_m(s_t, a_t))$. Once the recommendation system takes action a_t at state s_t, it will get the reward $r(s_t, a_t)$ in accordance with the user's feedback.
- **Discount Factor γ:** The discount factor in response to future rewards.

The goal of RL is to maximize the expected reward, so we define the discounted cumulative reward of the vector values to be $R_t = (R_{t,1}, ..., R_{t,m})$, in which $R_{t,m} = \sum_{t'=t}^{T} \gamma^{t'-t} \cdot r_m(s'_t, a'_t)$ is the cumulative rewards for discounts for individual feedback signal and T is the session length such as the number of requests between user and recommendation system. The state value function is the expected reward given the initial state, and its value is $V^\pi(s) = (V_1^\pi(s), ..., V_m^\pi(s)) = \mathbb{E}_\pi[R_t|s_t = s]$. If the action is given simultaneously, the Q-value function is the state value function with the value of $Q(s, a) = (Q_1^\pi(s, a), ..., Q_m^\pi(s)) = \mathbb{E}_\pi[R_t|s_t = s, a_t = a]$. Where π is a trainable policy function, the ultimate goal of RL is to obtain a trained policy function that maximizes the expected reward given the initial state to solve the following optimization problem:

$$\max_\pi \mathbb{E}_\pi[R_t|s_t = s] \iff \max_\pi \mathbb{E}_\pi[V^\pi(s)] \tag{1}$$

3.2. Model-Free Offline RL in Re-ranking

In model-free offline RL, we sample one trajectory at a time within a session where the user interacts with the recommendation system. Each trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, ..., s_T,$

$a_T, r_T)$ consists of a series of states, actions, and rewards. The reward returned at each timestamp t within the trajectory is the cumulative sum of all rewards from the current timestamp $R_t^\tau = \sum_{t'=t}^{T} r_{t'}^\tau$. Based on Equation 1, the objective of offline RL is to maximize the sum of rewards $\mathbb{E}_\tau[\sum_{t=1}^{T} r_t^\tau]$ which starting from the initial timestamp of the trajectory.

In offline RL, we cannot acquire data through user interaction with the recommendation system, but can only sample trajectories in the dataset with a fixed distribution of the action space. Thus there is the problem of extrapolation error where the distribution between environment and sample has bias. Decision-Transformer can correct bias through historical interactions in offline RL, so in this paper, we choose to use decision transformer to process the sequential states.

4. METHOD

In e-commerce scenarios, each time a user slides or turns a page is equivalent to a request, and the recommendation system will regenerate a new list of items based on the user's clicks on the previous page. Ideally, the recommendation system should be able to stimulate the user to keep sliding or turning the page, and at the same time, generate more clicks on the items to ultimately transform into the revenues of the e-commerce app, therefore, the user's attentions between different pages as well as attention on the different items on the same page all have sequential relationship, inspired by SPGA Feng et al. (2021a), we define it as global level (page) and local level (item) attention, there have been many studies on inter-item attention at the local level, but there are fewer studies on global page level attention in e-commerce scenarios, and to the best of our knowledge, we are the first study to consider the page-level attention in RL-based re-ranking.

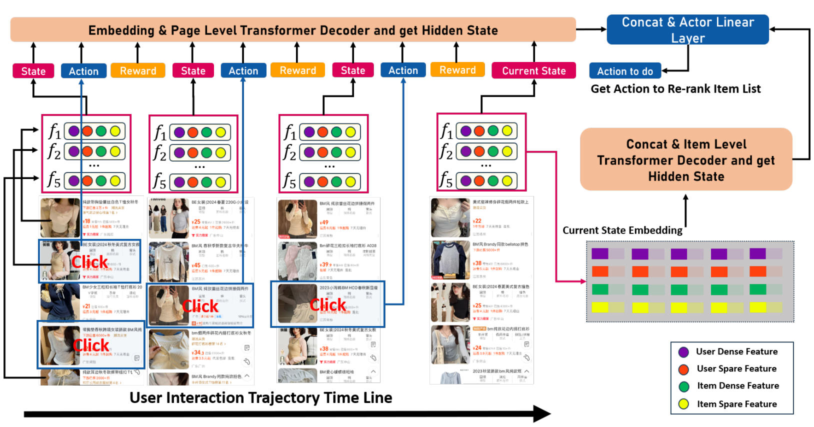

Figure 2: Decision transformer in page-level and item-level sequential decision. The page-level transformer decoder is employed for processing interaction trajectory, and the item-level transformer decoder processes the item list on the current page, with the final hidden layer's output corresponding to the state as input to the Actor.

4.1. Supervised Learning and Pre-train Embedding

The state inputs to PISDR have sparse and dense features for each item in the initial sorted list, sparse and dense features for the user, and sparse features for the user's history of clicking on items. We use the embedding layer to obtain low-dimensional dense layer embedding vectors from the corresponding sparse features. Each feature contains different attributes, e.g., the dense feature of an item contains the price of the item, the sparse feature contains it's id information, and the category information, etc., defining $m^i_{dense,j}$ as the j^{th} dense feature of the i^{th} item and $m^i_{spare,j}$ as the j^{th} sparse feature of the i^{th} item. The sparse features are converted to low-dimensional dense features through a learnable embedding layer matrix to obtain the embedding vector $e_{m,j} \in \mathbb{R}^{d^m_{e,j}}$ with low-dimensional dimension $d^m_{e,j}$ in which j represents the j^{th} sparse feature of item. After that, we put together the low-dimensional dense features obtained from the sparse features through the embedding layer with the original dense features of the items to obtain the embedding layer vector of the original sorted list of items:

$$\mathbf{x}_m = [e_{m,1} \bigoplus ... \bigoplus e_{m,N_{m,spare}} \bigoplus m_{dense,1} \bigoplus ... \bigoplus m_{dense,N_{m,dense}}], \tag{2}$$

in which $N_{m,spare}$ and $N_{m,dense}$ are the number of a item's sparse feature and dense feature, respectively.

Similarly, the input of the user features after the embedding layer is:

$$\mathbf{x}_u = [e_{u,1} \bigoplus ... \bigoplus e_{u,N_{u,spare}} \bigoplus u_{dense,1} \bigoplus ... \bigoplus u_{dense,N_{u,dense}}], \tag{3}$$

in which $N_{u,spare}$ and $N_{u,dense}$ are the number of the user's sparse feature and dense feature.

The embedding layer matrices of users and items are shared with the embedding layers of historical items as well as of the Actor-Multi-Tower-Critic network, and we pre-train the embedding layer parameters using a supervised learning approach. The loss of supervised learning is:

$$\mathbf{L}_{sup} = -\sum_{n=1}^{N} y_n \cdot log(p_n), \tag{4}$$

where $y_n \in \{0, 1\}$ is the click label of the n^{th} item in the initial ranking list and the $p_n \in \mathbb{R}^N$ is the output probability generated from the supervised learning model:

$$p = [p_1, ..., p_n] = MLP_{sup}(\mathbf{x}_m, \mathbf{x}_u), \tag{5}$$

Afterward, the embedding layer matrix can be updated by log-loss function (4).

4.2. Sequential Transformer Decision-Making

4.2.1. Global page-level sequential decision

In e-commerce scenarios, every time a user slides down or goes to the next page, a request is sent to the recommendation system to generate a new sorted sequence of recommended items. From a global perspective, the whole interaction process is that the user enters the next page, and the recommendation system generates a sorted sequence of items based on the user's features, and the clicked items in history. Therefore, inspired by the Decision Transformer, we believe that the user's feedback on the previously recommended item sequences can also be used as input to the model for the generation of the next re-ranking sequence.

Thus after getting the features of the user and item through the embedding layer, at the timestamp t, we use the information from the previous interactions to construct a trajectory to represent the T length time features following the form:

$$\tau_t =< a_{t-T}, R_{t-T}, S_{t-T+1}, a_{t-T+1}, R_{t-T+1}, ..., a_{t-1}, R_{t-1}, S_t >, \tag{6}$$

where T is the total length of the session, $S_t = MLP(Concat(\mathbf{x}_m.\mathbf{x}_u))$ is the fusion of user and item features at the timestamp t.

After constructing the trajectory, we feed the last K timesteps into the transformer decoder as Figure (2) shows, we learned three embeddings which are action embedding, reward embedding, and state embedding. After generating the three embeddings, we add page position embedding to the input of each embedding layer, after which the three output vectors are concatenated and processed using a transformer decoder to obtain the decoded vector. This process can also use a GPT model to generate the decoder vector, we believe that deep features can be more fully presented through the GPT. Then we use a linear layer to get the re-ranking list for the latest state.

The input of the global level transformer decoder at timestamp t is:

$$X_{t,global} \in \mathbb{R}^{3*(T-1)\times dim} = \{Emb(a_{t-T}), Emb(R_{t-T}), ..., Emb(S_t)\}, \tag{7}$$

where Emb is the function of the embedding, dim is the dimension of embedding output.

Define the output of i_{th} layers multi attention is:

$$A_{i,global} = MultiAttention(Q_{i-1}, K_{i-1}, V_{i-1}) = Concat(h_1, ..., h_H) \tag{8}$$

$$h_i = Attention(q_i, k_i, v_i) = softmax(\frac{q_i k_i^T}{\sqrt{dim}})v_i, \tag{9}$$

in which $Q_{i-1}, K_{i-1}, V_{i-1} = A_{i-1,global} \times (W_Q, W_K, W_V)$, $A_{0,global} = X_{t,global}$ and $W_Q, W_K, W_V \in \mathbb{R}^{dim\times dim}$ is the weight matrix of query, key and value.

The final output of the global page-level sequential decision is the last dimension of the hidden state which means use the hidden layer state of the L_{th} layer of S_T as input to the Actor :

$$X_{out,state} = MultiAttention(A_{L,global})[-1]. \tag{10}$$

4.2.2. Local item-level sequential decision

According to the initial item list, we model the sequential decision at the item level, and output the re-ranking action sequence through the linear layer.

The input of the local level transformer decoder is:

$$X_{t,local} \in \mathbb{R}^{N\times dim} = \{Emb(I_1), Emb(I_2), ..., Emb(I_N)\}, \tag{11}$$

where $I_i = Concat(I^i_{spare}, I^i_{dense})$ is the fusion of the i_{th} item spare and dense feature.

Similar to global sequential decision, the final output of the local item-level sequential decision is:

$$X_{out,item} = MultiAttention(A_{L,local})[-1]. \tag{12}$$

After obtaining the hidden layer output $X_{out,state}$ and $X_{out,item}$ at the global and local levels, they are fused and fed into the Actor network to generate the re-ranking sequence, the re-ranking score at timestamp t is:

$$a_t = Sigmoid(MLP(Concat(X_{out,state}, X_{out,item}))). \tag{13}$$

4.3. Multi-Tower Critic for Multi-Feedback Metrics

Inspired by existing work RMTL Zhuang et al. (2018), we use multi-tower critic as shown in Figure (3) to integrate these three feedback metrics in order of priority, using click-through rate as the primary evaluation metric, conversion rate, and fine-tuning score as the secondary metrics, and calculating the Q-value for updating the actor network using a weighted approach. For m feedback metrics, we have m different critics, defining the

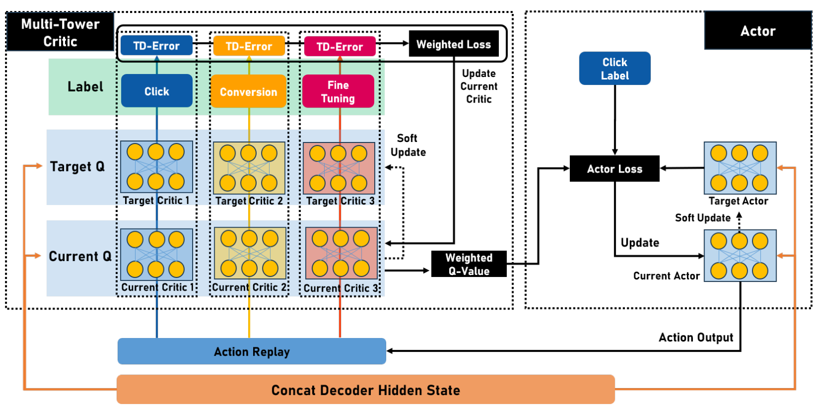

Figure 3: Actor with Multi-Tower Critic. Evaluating Multiple Feedback Metrics Using Multi Tower Critic, Updating Actor Networks Using Weighted Values.

$k^{th}, k \in [1, m]$ assessment metric with a Q value of:

$$Q_k(s_t, a_t) = \mathbb{E}[r_k(s_t, a_t) + \gamma V(s_{t+1}) | s_t, a_t], \tag{14}$$

in which, a_t is the output re-ranking score from actor network Equ.(13) and we define the weighted value of each feedback indicator to be ω_k, the td-error of the k^{th} critic is:

$$TD_k = r_k(s_t, a_t) + \gamma Q(s_{t+1}, a_{t+1}; \tilde{\phi_k}) - Q(s_t, a_t; \phi_k), \tag{15}$$

where ϕ_k is the parameters of the k^{th} current critic network, and $\tilde{\phi_k}$ is the parameters of the k^{th} target critic network. Then we update the k^{th} current critic network for each task by the following gradient descent:

$$\phi_k = \phi_k - \gamma_\phi \nabla TD_k. \tag{16}$$

Then we define the weighted Q value at t^{th} timesteps is: $Q_{\omega,t} = \sum_{k=1}^{m} \omega_k Q_k(s_t, a_t | \phi_k)$ We update the actor network by minimizing the loss function:

$$\mathbf{L} = \gamma_\theta Q_{\omega,t} + \sum_{\tau} Logloss(\pi(s_t; \theta_k), y_t) \tag{17}$$

in which y_t is the label of the click in the item list. Finally, we update the target network every t timesteps:

$$\tilde{\theta_k} = \beta * \tilde{\theta_k} + (1 - \beta)\theta_t \tag{18}$$

$$\tilde{\phi} = \beta * \tilde{\phi} + (1 - \beta)\phi \tag{19}$$

5. EXPERIMENTS

In this section, we conduct several experiments using a real world public dataset *Avito*, and the online industry dataset to evaluate the effectiveness of our framework.

5.1. Experimental Setup

5.1.1. Dataset

- **Avito** [1] The public dataset comprises user search logs and metadata from avito.ru. To organize it in the form of a session level, we pre-process the entire dataset by clustering it according to the user-id information and filtering out the advertisement click information corresponding to user IDs with less than M total clicks.

- **1688**. The online 1688 industrial dataset, containing daily user clicks on the main product promotion screen, is accessed three million times a day. Similar to the processing with Avito, we cluster the 1688 dataset by users and timestamps and filter the information with more than M total clicks, which contains the user's profile, the information of the product, and the user's historical click sequence.

5.1.2. Evaluation Metrics

Our proposed model and baselines undergo evaluation using both ranking and utility metrics. Regarding ranking metrics, we utilize the widely accepted MAP@K and NDCG@K metrics, consistent with prior studies. Specifically, we employ the prevalent AUC metric to assess the effectiveness of the prediction module.

5.1.3. Initial ranker and baselines

We use the **LambdaMART** to generate initial lists. **LambdaMART** is a state-of-the-art listwise learning-to-rank algorithm, which optimizes NDCG directly. To compare the proposed model with the following state-of-the-art reranking models, listed as follows:

- **PRM**(Pei et al. (2019)): employs self-attention to model the mutual influence between any pair of items and users' preferences.

- **SetRank**(Pang et al. (2020)): learns permutation-equivariant representations for the inputted items via self-attention.

- **DLCM**(Ai et al. (2018)): first apply GRU to encode and rerank the top results.

1. https://www.kaggle.com/c/avito-context-ad-clicks/overview.

- **CMR**(Chen et al. (2023)): a rerank model to adapt the recommendation re-ranking models according to the preference weights in a dynamic manner.
- **MIR**(Xi et al. (2022)): a rerank model can estimate the reranking score on the ordered initial list before reranking.
- **EGRerank**(Du et al. (2018)): a rank model adopts an evaluator generator paradigm.
- **PRS (Beam-Search & Evaluator)**(Feng et al. (2021a)): uses the beam search method to generate K candidate lists based on the calculated estimated reward and a evaluator to rank multiple candidate lists.

5.1.4. Implementation details

All experiments use mini-batches of 256 training examples and the Transformer Decoder with 512 hidden units. We train PISDR with the Adam optimizer with a learning rate of 0.0001. We regularize the decoder model by using dropout with the probability of 0.1.

5.2. Offline Experiments

We conducted a performance comparison of PISDR [2] with six other models, assessing their performance based on MAP, NDCG, and AUC. The results are presented in Table 1.

Table 1: Offline evaluation results on Avito dataset and 1688 dataset (bold: best).

	Avito							1688						
	MAP			NDCG			AUC	MAP			NDCG			AUC
	@3	@5	@12	@3	@5	@12	/	@5	@10	@12	@5	@10	@12	/
PRMPei et al. (2019)(2019)	0.3974	0.4506	0.4694	0.4040	0.4849	0.5731	0.6886	0.4261	0.4410	0.4408	0.4598	0.5774	0.5865	0.6045
SetRankPang et al. (2020)(2020)	0.4001	0.4537	0.4720	0.4368	0.5488	0.6029	0.6905	0.4144	0.4325	0.4326	0.4468	0.5690	0.5788	0.5806
DLCMAi et al. (2018)(2018)	0.3987	0.4544	0.4714	0.4375	0.5564	0.6029	0.6868	0.4254	0.4404	0.4403	0.4589	0.5765	0.5859	0.6023
CMRChen et al. (2023)(2023)	0.3092	0.3241	0.3669	0.3269	0.3552	0.5143	0.5607	0.4142	0.4324	0.4325	0.4479	0.5687	0.5787	0.5844
MIRXi et al. (2022)(2022)	0.3938	0.4474	0.4669	0.4314	0.5445	0.5990	0.6889	0.4178	0.4350	0.4351	0.4555	0.5717	0.5809	0.6015
EGRerankDu et al. (2018)(2018)	0.4026	0.4426	0.4618	0.4391	0.5034	0.5915	0.6915	0.4173	0.4347	0.4348	0.4537	0.5714	0.5807	0.6012
PRSFeng et al. (2021a)(2021)	/	/	/	/	/	/	/	0.4246	0.4395	0.4392	0.4565	0.5755	0.5852	0.6020
PISDR(Ours)	**0.4041**	**0.4608**	**0.4767**	**0.4428**	**0.5639**	**0.6069**	**0.6943**	**0.4264**	**0.4412**	**0.4410**	**0.4606**	**0.5782**	**0.5874**	**0.6054**

Our proposed PISDR model outperforms state-of-the-art methods across all metrics. As illustrated in Table 1, PISDR attains the highest scores in re-ranking metrics including MAP, NDCG, and AUC. Specifically, in terms of the MAP metric, PISDR exhibits a 0.3%-1.4% improvement over the EGRerank and DLCM. Furthermore, PISDR outperforms the EGRerank, DLCM and SetRank by 0.6%-1.3% on the NDCG@ metric.

Given that PISDR attains superior results in both MAP and NDCG metrics, it can be argued that sequential decision plays a significant role in effectively addressing the challenge of extrapolation errors in offline RL. In terms of AUC, PISDR and EGRerank achieve higher performance than others, therefore the use of RL can achieve better performance than SL in recommendation systems.

On the 1688 dataset, we conducted a performance comparison of PISDR with seven other models across three key aspects: MAP, NDCG, and AUC. Our proposed PISDR model consistently outperforms state-of-the-art methods in all evaluated metrics. As indicated in Table 1, in the context of the MAP@5 metric, PISDR exhibits a 0.1% improvement over the PRM, along with similar improvements of 0.1% for MAP@10 and MAP@12. In terms of

2. The code is publicly accessible at https://anonymous.4open.science/r/PISDR-832B.

NDCG, PISDR achieves a 0.2% enhancement over PRM for NDCG@5, 0.1% for NDCG@10, and 0.2% for NDCG@12. Moreover, PISDR demonstrates a 0.1% improvement over PRM in the overall AUC metric.

The DLCM and PRM perform more significantly this is due to the 1688 dataset containing a higher proportion of session-level data compared to the Avito dataset. PRM contains more attention mechanisms compared to DLCM and achieves better results due to its ability to adapt to this data distribution. Additionally, the uneven distribution of users in the 1688 dataset, with differences between users from the previous day and current day, presents challenges for evaluators trained by EGRerank. Consequently, the impact of re-ranking, as assessed by EGRerank, is less pronounced compared to that observed in the Avito dataset. PISDR, in comparison to PRM, introduces page-level attention considerations and incorporates item-level attention mechanisms into the decision-making process within the current state. This approach allows for the extraction of user interests from a dynamic, sliding perspective, thereby further enhancing the re-ranking effectiveness.

5.3. Ablation Study

The most essential modules of the PISDR model are the pre-train embedding (SLPE), Global page-level sequential decision (GPSD) and local item-level sequential decision. To explore the effectiveness of these modules in PISDR, we conduct ablation studies on Avito dataset. All experiments were repeated 3 times and the averaged AUC is in Table (2).

Table 2: Result of ablation experiment of re-ranking model on Avito dataset.

	MAP			NDCG			AUC
	@3	@5	@12	@3	@5	@12	/
PISDR	0.4041	0.4608	0.4767	0.4428	0.5639	0.6069	0.6943
-SLPE	0.3891	0.4453	0.4620	0.4293	0.5482	0.5958	0.6839
-GPSD	0.3909	0.4472	0.4651	0.4290	0.5487	0.5979	0.6912
-STDM	0.3869	0.4422	0.4600	0.4274	0.5434	0.5942	0.6803

PISDR (-SLPE) blocks the supervised learning pre-training embedding layer. This particular layer is designed for the pre-training of item information, which can subsequently be utilized in the Actor-Critic network. As demonstrated in Table 3, MAP@k decreases by 0.0150/0.0175/0.0147, NDCG@k decreases by 0.0135/0.0157/0.0111, AUC decreases by 0.0104, suggesting that it is more appropriate to pretrain item features rather than randomly generating them within the RL network. PISDR (-GPSD) blocks the GPSD module and only keeps the local item-level sequential decision. For instance, when k=3, 5, and 10, MAP@k decreases by 0.0132/0.0136/0.0116, respectively. Similarly, NDCG@k decreases by 0.0138/0.0152/0.0090, respectively. Additionally, the AUC decreases by 0.0031. These findings highlight the significance of extracting context information from previous pages, which is effectively addressed by the proposed GPSD module. To further investigate the complementary roles of page-level and item-level sequential decision mechanisms, we blocks the total Sequential Transformer Decision-Making (STDM) module. As demonstrated in Table 3, PISDR (-GPSD) outperforms PISDR (-STDM) across multiple metrics, namely MAP@k, NDCG@k, and AUC. The last experiment demonstrates the efficacy of the combined utilization of page-level and item-level attention modules.

5.4. Online Experiments

Table 3: Online A/B test results.

Model	CTR	content exposure number per user
Base Ranking Model	+0.00%	+0.00%
PRS	+1.56%	+2.02%
PISDR	+1.96%	+2.59%

As in CMRChen et al. (2023) and PIERShi et al. (2023), the online A/B test uses the online baseline model to compare with PISDR. We compare PISDR with the base ranking model which is similar as Deep Match to Rank ModelWei et al. (2023) and PRS and all deployed on the recommended scenes on the homepage of 1688 APP through online A/B test. Specifically, we will conduct a two-week online A/B test in July 2023 using 5% of the total production traffic. As a result, we find that PISDR gets CTR and content exposure number per user increase by 1.96% and 2.59% respectively. Compared with PRS, PISDR has also improved CTR by 0.40% and content exposure number per user by 0.57% .

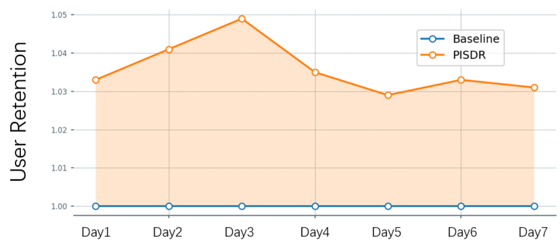

Figure 4: Online A/B test on return visit rate in five days.

Finally, we compared the percentage of improvement in user retention between PISDR and the baseline over a 7-day period, as shown in Figure 4. This suggests that PISDR can better capture the user's interest and is more effective in re-ranking recommendations.

6. CONCLUSION

In e-commerce app, a significant portion of re-ranking tasks is currently reliant on supervised learning, often lacking the optimization of long-term benefits. We propose an offline RL method aimed at optimizing three crucial metrics: MAP, NDCG, and AUC, particularly on session-level datasets. Our approach takes into account user scroll-down actions, introduces a fusion of page-level and item-level attention mechanisms, and leverages a decision transformer methodology to mitigate extrapolation errors associated with offline RL. Experimental results demonstrate that incorporating sequential decision-making contributes to a noticeable enhancement in model performance. Through online A/B testing, our proposed framework leads to a substantial 2.59% increase in CTR.

References

M Mehdi Afsar, Trafford Crump, and Behrouz Far. Reinforcement learning based recommender systems: A survey. *ACM Computing Surveys*, 55(7):1–38, 2022.

Qingyao Ai, Keping Bi, Jiafeng Guo, and W. Bruce Croft. Learning a deep listwise context model for ranking refinement. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, pages 135–144, New York, NY, USA, 2018. ACM.

Irwan Bello, Sayali Kulkarni, Sagar Jain, Craig Boutilier, Ed Chi, Elad Eban, Xiyang Luo, Alan Mackey, and Ofer Meshi. Seq2slate: Re-ranking and slate optimization with rnns, 2018.

Qingpeng Cai, Zhenghai Xue, Chi Zhang, Wanqi Xue, Shuchang Liu, Ruohan Zhan, Xueliang Wang, Tianyou Zuo, Wentao Xie, Dong Zheng, Peng Jiang, and Kun Gai. Two-stage constrained actor-critic for short video recommendation. In *Proceedings of the ACM Web Conference 2023*, WWW '23, pages 865–875, New York, NY, USA, 2023. ACM.

Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. Top-k off-policy correction for a reinforce recommender system. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 456–464, 2019.

Sirui Chen, Yuan Wang, Zijing Wen, Zhiyu Li, Changshuo Zhang, Xiao Zhang, Quan Lin, Cheng Zhu, and Jun Xu. Controllable multi-objective re-ranking with policy hypernetworks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23', pages 3855–3864, New York, NY, USA, 2023. ACM.

Lei Du, Yan Pan, Jintang Ding, Hanjiang Lai, and Changqin Huang. Egrank: an exponentiated gradient algorithm for sparse learning-to-rank. *Information Sciences*, 467:342–356, 2018.

Yufei Feng, Yu Gong, Fei Sun, Junfeng Ge, and Wenwu Ou. Revisit recommender system in the permutation prospective. *arXiv preprint arXiv:2102.12057*, 2021a.

Yufei Feng, Yu Gong, Fei Sun, Junfeng Ge, and Wenwu Ou. Revisit recommender system in the permutation prospective, 2021b.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019.

Chongming Gao, Kexin Huang, Jiawei Chen, Yuan Zhang, Biao Li, Peng Jiang, Shiqi Wang, Zhong Zhang, and Xiangnan He. Alleviating matthew effect of offline reinforcement learning in interactive recommendation. In *SIGIR '23*, pages 238–248, New York, NY, USA, 2023a. ACM.

Chongming Gao, Shiqi Wang, Shijun Li, Jiawei Chen, Xiangnan He, Wenqiang Lei, Biao Li, Yuan Zhang, and Peng Jiang. Cirs: Bursting filter bubbles by counterfactual interactive recommender system. *ACM Transactions on Information Systems*, 42(1):1–27, 2023b.

Olivier Jeunen and Bart Goethals. Pessimistic reward models for off-policy learning in recommendation. In *Proceedings of the 15th ACM Conference on Recommender Systems*, pages 63–74, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Yi Li, Jieming Zhu, Weiwen Liu, Liangcai Su, Guohao Cai, Qi Zhang, Ruiming Tang, Xi Xiao, and Xiuqiang He. Pear: Personalized re-ranking with contextualized transformer for recommendation. In *Companion Proceedings of the Web Conference 2022*, pages 62–66, New York, NY, USA, 2022. ACM.

Yuanguo Lin, Yong Liu, Fan Lin, Lixin Zou, Pengcheng Wu, Wenhua Zeng, Huanhuan Chen, and Chunyan Miao. A survey on reinforcement learning for recommender systems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

Feng Liu, Ruiming Tang, Xutao Li, Weinan Zhang, Yunming Ye, Haokun Chen, Huifeng Guo, Yuzhou Zhang, and Xiuqiang He. State representation modeling for deep reinforcement learning based recommendation. *Knowledge-Based Systems*, 205:106170, 2020a.

Weiwen Liu, Qing Liu, Ruiming Tang, Junyang Chen, Xiuqiang He, and Pheng Ann Heng. Personalized re-ranking with item relationships for e-commerce. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 925–934, New York, NY, USA, 2020b. ACM.

Ziru Liu, Jiejie Tian, Qingpeng Cai, Xiangyu Zhao, Jingtong Gao, Shuchang Liu, Dayou Chen, Tonghao He, Dong Zheng, Peng Jiang, and Kun Gai. Multi-task recommendations with reinforcement learning. In *Proceedings of the ACM Web Conference 2023*, WWW '23, pages 1273–1282, New York, NY, USA, 2023. ACM.

Liang Pang, Jun Xu, Qingyao Ai, Yanyan Lan, Xueqi Cheng, and Jirong Wen. Setrank: Learning a permutation-invariant ranking model for information retrieval. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pages 499–508, 2020.

Changhua Pei, Yi Zhang, Yongfeng Zhang, Fei Sun, Xiao Lin, Hanxiao Sun, Jian Wu, Peng Jiang, Junfeng Ge, Wenwu Ou, and Dan Pei. Personalized re-ranking for recommendation. In *RecSys '19*, pages 3–11, New York, NY, USA, 2019. ACM.

Xiaowen Shi, Fan Yang, Ze Wang, Xiaoxu Wu, Muzhi Guan, Guogang Liao, Wang Yongkang, Xingxing Wang, and Dong Wang. Pier: Permutation-level interest-based end-to-end re-ranking framework in e-commerce. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23', pages 4823–4831, New York, NY, USA, 2023. ACM.

Aditya Srinivas Timmaraju, Mehdi Mashayekhi, Mingliang Chen, Qi Zeng, Quintin Fettes, Wesley Cheung, Yihan Xiao, Manojkumar Rangasamy Kannadasan, Pushkar Tripathi, Sean Gahagan, et al. Towards fairness in personalized ads using impression variance aware reinforcement learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4937–4947, 2023.

D. Xiao Wang and T. & Wang. A general offline reinforcement learning framework for interactive recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 2021.

Siyu Wang, Xiaocong Chen, Dietmar Jannach, and Lina Yao. Causal decision transformer for recommender systems via offline reinforcement learning. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1599–1608, 2023.

Yuyan Wang, Mohit Sharma, Can Xu, Sriraj Badam, Qian Sun, Lee Richardson, Lisa Chung, Ed H Chi, and Minmin Chen. Surrogate for long-term user experience in recommender systems. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 4100–4109, 2022.

Liuwei Wei, Xi Yunjia, Qin Jiarui, Sun Fei, Chen Bo, Zhang Weinan, Zhang Rui, and Tang Ruiming. Neural re-ranking in multi-stage recommender systems: A review. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence Survey Track*, pages 5512–5520, 2022.

Penghui Wei, Yongqiang Chen, Shaoguo Liu, Liang Wang, and Bo Zheng. Rltp: Reinforcement learning to pace for delayed impression modeling in preloaded ads. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5204–5214, 2023.

Yunjia Xi, Weiwen Liu, Jieming Zhu, Xilong Zhao, Xinyi Dai, Ruiming Tang, Weinan Zhang, Rui Zhang, and Yong Yu. Multi-level interaction reranking with user behavior history. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pages 1336–1346. ACM, 2022.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.

Tao Zhuang, Wenwu Ou, and Zhirong Wang. Globally optimized mutual influence aware ranking in e-commerce search. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 3725–3731. International Joint Conferences on Artificial Intelligence Organization, 2018.

Lixin Zou, Long Xia, Pan Du, Zhuo Zhang, Ting Bai, Weidong Liu, Jian-Yun Nie, and Dawei Yin. Pseudo dyna-q: A reinforcement learning framework for interactive recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 816–824, 2020.