## Deep Reinforcement Learning based Recommendation with Explicit User-Item Interactions Modeling

## 基于深度强化学习的推荐，具有明确的用户-项目交互建模

Abstract—Recommendation is crucial in both academia and industry, and various techniques are proposed such as contentbased collaborative filtering, matrix factorization, logistic regression, factorization machines, neural networks and multiarmed bandits. However, most of the previous studies suffer from two limitations: (1) considering the recommendation as a static procedure and ignoring the dynamic interactive nature between users and the recommender systems; (2) focusing on the immediate feedback of recommended items and neglecting the long-term rewards. To address the two limitations, in this paper we propose a novel recommendation framework based on deep reinforcement learning, called DRR. The DRR framework treats recommendation as a sequential decision making procedure and adopts an “Actor-Critic” reinforcement learning scheme to model the interactions between the users and recommender systems, which can consider both the dynamic adaptation and longterm rewards. Further more, a state representation module is incorporated into DRR, which can explicitly capture the interactions between items and users. Three instantiation structures are developed. Extensive experiments on four real-world datasets are conducted under both the offline and online evaluation settings. The experimental results demonstrate the proposed DRR method indeed outperforms the state-of-the-art competitors. Index Terms—Recommendation, Deep Reinforcement Learning, User-Item Interactions

摘要： 推荐在学术界和工业界都是一个非常重要的问题，各种推荐技术被提出，如基于内容的协同过滤、矩阵分解、逻辑回归、分解机、神经网络和多臂赌博机等。然而，以往的研究大多存在两个局限性:

(1)将推荐视为一个静态过程，忽视了用户与推荐系统之间的动态交互特性;

(2)注重被推荐物品的即时反馈，忽视长期奖励。

针对这两个局限性，本文提出了一种基于深度强化学习的推荐框架DRR。DRR框架将推荐视为一个顺序决策过程，采用“行动器-评判器”强化学习方案来模拟用户与推荐系统之间的交互，既考虑动态变化，又考虑长期奖励。在DRR中加入了一个状态表示模块，可以显式地捕获项目与用户之间的交互。开发了三种实例化结构。在离线和在线评估设置下，在四个真实世界的数据集上进行了大量的实验。实验结果表明，所提出的DRR方法确实优于最先进的竞争对手。

索引-推荐，深度强化学习，用户-项目交互

1. INTRODUCTION

Thanks to the increasing online services, such as online shopping, online news and online social networks, it becomes quite convenient to acquire items (goods, books, videos, news, etc.) via Internet or mobile devices. Albeit the great convenience, the overwhelming number of items in the systems also pose a significant challenge for users, to find the items that match their interests. Recommendation is a widely used solution and various families of techniques have been proposed, such as content-based collaborative filtering [1], matrix factorization based methods [2]–[5], logistic regression, factorization machines and its variants [6]–[8], deep learning models [9]–[12] and multi-armed bandits [13]–[17]. However, such mentioned studies suffer from two serious limitations.

由于在线购物，在线新闻和在线社交网络等在线服务的增加，通过Internet或移动设备获取商品（商品，书籍，视频，新闻等）变得非常方便。 尽管提供了极大的便利，但是系统中大量的项目也给用户带来了很大的挑战，即要找到与他们的兴趣相匹配的项目。 推荐是一种广泛使用的解决方案，并且已经提出了多种技术，例如基于内容的协作过滤[1]，基于矩阵分解的方法[2] – [5]，逻辑回归，分解机及其变体[6] – [8]，深度学习模型[9]-[12]和多臂赌博机[13]-[17]。 但是，这些提到的研究有两个严重的局限性。

Firstly, most of them consider the recommendation procedure as a static process, i.e., they assume the user’s underlying preference keeps unchanged. However, it is very common that a user’s preference is dynamic w.r.t. time, i.e., a user’s preference on previous items will affect her choice on the next items. Hence, it would be more reasonable to model the recommendation as a sequential decision making process. We will show some evidence observed in publicly available datasets (MovieLens and Yahoo! Music) to support our opinion. In the two datasets, the sequential behaviors of users are recorded and we are interested in what would happen if a user consecutively receives satisfied or unsatisfied recommendations. Though the datasets do not record any recommendation procedure, we can simulate this according to the users’ ratings, namely, consecutive rating “positive” (“negative”) simulates that a user consecutively receives satisfied (unsatisfied) recommendations. As presented in Figure 1, we observe that a user tends to gives a higher (lower) rating if she has consecutively received more satisfied (unsatisfied) items, as shown by the green (red) line, where the blue dot line denotes the average rating for reference. This suggests that a user will be more pleasant (unpleasant) if she consecutively receives more satisfied (unsatisfied) recommendations and therefore she tends to give a higher (lower) rating to the current recommendation. Hence, the user’s dynamic preference suggests that a good recommendation should be modeled as a sequential decision making process.

首先，大多数人认为推荐程序是一个静态过程，即他们假设用户的基本偏好保持不变。但是，用户的偏好是动态的，这是很常见的。时间，即用户对上一个项目的偏好会影响她对下一个项目的选择。因此，将建议建模为顺序决策过程将更为合理。我们将展示一些在公开可用的数据集（MovieLens和Yahoo! Music）中观察到的证据，以支持我们的观点。在这两个数据集中，记录了用户的顺序行为，我们对如果用户连续收到满意或不满意的建议会发生什么感兴趣。尽管数据集没有记录任何推荐过程，但我们可以根据用户的评分进行模拟，即连续评分“正”（“负”）模拟用户连续收到满意（不满意）的推荐。如图1所示，我们观察到，如果用户连续收到更多满意（不满意）的商品，则趋向于给出较高（较低）的评分，如绿色（红色）线所示，其中蓝色虚线表示平均值评级以供参考。这表明，如果用户连续收到更满意（不满意）的推荐，将会更加愉快（不愉快），因此，她倾向于对当前推荐给予更高（更低）的评分。因此，用户的动态偏好表明，一个好的推荐应该被建模为一个顺序决策过程。

Secondly, the aforementioned studies are trained by maximizing the immediate rewards of recommendations, which merely concentrates on whether the recommended items are clicked or consumed, but ignores the long-term contributions that the items can make. However, the items with small immediate rewards but large long-term benefits are also crucial [18]. We take an example in News recommendation [19] to explain this. As a user requests for news to read, two possible pieces of news may lead to the same immediate reward, i.e., the user will click and read the two pieces of news with equal probability, where one is about a thunderstorm alert and the other is about a basketball player Kobe Bryant. In this example, after reading the news about thunderstorm, the user probably is not willing to read news about this issue anymore; while on the other hand, the user will possibly read more about NBA or basketball after reading the news about Kobe. The fact suggests that recommending the news about Kobe will introduce more longterm rewards. Hence, when recommending items to users, both the immediate and long-term rewards should be taken into consideration.

其次，通过最大化推荐的即时回报来训练上述研究，该研究仅集中在点击或消费推荐的商品上，而忽略了这些商品可以做出的长期贡献。然而，立即回报少但长期利益大的项目也至关重要[18]。我们以新闻推荐[19]中的示例为例对此进行解释。当用户要求阅读新闻时，两条可能的新闻可能会导致相同的即时奖励，即，用户将以相等的概率单击并阅读这两条新闻，其中一条与雷暴警报有关，另一条与雷暴警报有关。关于篮球运动员科比·布莱恩特（Kobe Bryant）。在此示例中，在阅读了有关雷暴的新闻之后，用户可能不再愿意阅读有关此问题的新闻；而另一方面，用户在阅读有关科比的新闻后可能会阅读有关NBA或篮球的更多信息。事实表明，推荐有关科比的新闻将带来更多的长期回报。因此，在向用户推荐物品时，应同时考虑到近期和长期的奖励。

Recently, Reinforcement Learning (RL) [20], which has shown great potential in various challenging scenarios that require both dynamic modeling and long term planning, such as game playing [21], [22], real-time ads bidding [23], [24], neural network structure searching [25], [26], is introduced in recommender systems [18], [19], [27]–[33].

最近，强化学习(RL)[20]被引入到推荐系统[18]、[19]、[27]-[18]0中，它在各种需要动态建模和长期规划的具有挑战性的场景中显示出了巨大的潜力，如玩游戏[21]、[22]、实时广告竞价[23]、[24]、神经网络结构搜索[25]、[26]等。

In the early stage, model-based RL techniques are proposed to model recommendation procedure, such as POMDP [18] and Q-learning [27]. However, these methods are inapplicable to complicated recommendation scenarios when the number of candidate items is large, because a time-consuming dynamic programming step is required to update the model. Later, model-free RL techniques are utilized in recommender systems, from both academia and industry. Such techniques can be divided into two categories: value-based [19], [29] and policy-based [28], [32], [33]. Value-based approaches compute Q-values of all available actions for a given state and the one with the maximum Q-value is selected as the best action. Due to the evaluation on overall actions, the approaches may become very inefficient if the action space is too large. As for the policy-based approaches, this type of studies generate a continuous parameter vector as the representation of an action [28], [32], [33], which can be utilized in generating the recommendation and updating the Q-value evaluator. Thanks to the continuous representations, the inefficiency drawbacks can be overcome. However, these studies [28], [32], [33] still have one common limitation: the user state is learnt via a conventional fully connected neural network, which does not explicitly and carefully model the interactions between users and items.

在早期阶段，提出了基于模型的RL技术来对推荐程序进行建模，例如POMDP [18]和Q学习[27]。但是，当候选项目的数量很大时，这些方法不适用于复杂的推荐方案，因为更新模型需要耗时的动态规划步骤。后来，学术界和工业界的推荐系统都采用了无模型的RL技术。

这些技术可以分为两类：基于价值的[19]，[29]和基于策略的[28]，[32]，[33]。基于值的方法将计算给定状态下所有可用操作的Q值，并选择具有最大Q值的方法作为最佳操作。由于对总体行动进行了评估，如果行动空间太大，这些方法可能会变得效率很低。对于基于策略的方法，此类研究会生成一个连续的参数向量作为动作[28]，[32]，[33]的表示，可用于生成建议和更新Q值评估器。由于连续的表示，可以克服效率低下的缺点。但是，这些研究[28]，[32]，[33]仍然有一个共同的局限性：用户状态是通过常规的完全连接的神经网络学习的，该网络无法明确，仔细地建模用户与物品之间的交互。

In this paper, to break the limitations stated above, we propose a deep reinforcement learning based recommendation framework with explicit user-item interactions modeling (DRR). The “Actor-Critic” type framework DRR is incorporated with a state representation module, which explicitly models the complex dynamic user-item interactions to pursuit better recommendation performance. Specifically, the embeddings of users and items from the historical interactions are fed into a carefully designed multi-layer network, which explicitly models the interactions between users and items, to produce a continuous state representation of the user in terms of her underlying sequential behaviors. This network is named as the state representation module, which plays two important roles in our framework. On the one hand, it is utilized to generate an ranking action to calculate the recommendation scores for ranking. On the other hand, the state representation together with the generated action is the input of the Critic network, which aims to estimate the Q-value, i.e., the quality of the action in the current state. Based on the evaluation, the Actor (policy) network can be updated. We note that both the Actor and Critic networks are carefully designed by modeling the interactions between users and items explicitly. Extensive experiments on four real-world datasets demonstrate that the proposed method yields superior performance than the stateof-the-art methods. The main contributions of this paper can be summarized as follows:

在本文中，为了打破上述限制，我们提出了一种基于深度强化学习的推荐框架，其中包含显式的用户-项目交互建模（DRR）。 “ Actor-Critic”类型的框架DRR与状态表示模块结合在一起，该模块显式地对复杂的动态用户项交互进行建模，以追求更好的推荐性能。具体而言，将来自历史交互的用户和项目的嵌入内容输入到到精心设计的多层网络中，该网络明确地对用户和项目之间的交互进行建模，以根据用户的基础顺序行为来生成用户的连续状态表示。该网络被称为状态表示模块，在我们的框架中扮演两个重要角色。一方面，它被用来产生排名动作，以计算用于排名的推荐分数。另一方面，状态表示和所生成的动作是Critic网络的输入，该网络旨在估计Q值，即当前状态下动作的质量。基于评估，可以更新Actor（策略）网络。我们注意到，Actor和Critic网络都是通过对用户和项目之间的交互进行显式建模而精心设计的。在四个真实世界的数据集上进行的大量实验表明，与现有技术相比，该方法具有更好的性能。本文的主要贡献可归纳如下：

• We propose a deep reinforcement learning based recommendation framework DRR. Unlike the conventional studies, DRR adopts an “Actor-Critic” structure and treats the recommendation as a sequential decision making process, which takes both the immediate and long-term rewards into consideration.

1.我们提出了一个基于深度强化学习的推荐框架DRR。 与常规研究不同，DRR采用“Actor-Critic”结构，并将推荐视为顺序决策过程，该过程将即时和长期奖励都考虑在内。

• Under the DRR framework, three different network structures are proposed, which can explicitly model the interactions between users and items.

•在DRR框架下，提出了三种不同的网络结构，它们可以显式地建模用户与项目之间的交互。

Extensive experiments are carried out on four real-world datasets, and the results demonstrate the proposed methods indeed outperforms the state-of-the-art competitors.

在四个真实的数据集上进行了广泛的实验，结果表明，所提出的方法的确优于最先进的竞争对手。

The rest of this paper is organized as follows. Related work and background are presented in Section II. The preliminary knowledge is presented in Section III. The proposed methods are introduced in Section IV. Experimental details and results are discussed in Section V. Finally, we conclude this paper and discuss some future work in Section VI.

本文的其余部分安排如下。 第二部分介绍了相关的工作和背景。 初步知识在第三部分中介绍。 第四节介绍了建议的方法。 实验细节和结果将在第五节中讨论。最后，我们将对本文进行总结，并在第六节中讨论一些未来的工作。

II. RELATED WORK

1. Non-RL based Recommendation Techniques

Various kinds of recommendation techniques are proposed in the past a few decades to improve the performance of recommender systems, including content-based filtering [1], matrix factorization based methods [2]–[5], logistic regression, factorization machines and its variants [6]–[8], and until recently deep learning models [9]–[12].

在过去的几十年中，人们提出了各种推荐技术来改善推荐系统的性能，包括基于内容的过滤[1]，基于矩阵分解的方法[2] – [5]，逻辑回归，分解机及其变体。 [6] – [8]，以及直到最近的深度学习模型[9] – [12]。

At the beginning of this century, content-based filtering [1] is proposed to recommend items by considering the content similarity between items. Later, collaborative filtering (CF) is put forward and extensively studied. The rationale behind CF is that the users with similar behaviors tend to prefer the same items, and the items consumed by similar users tend to have the same rating. However, conventional CF based methods tend to suffer from the data scarcity, because the similarity calculated from sparse data can be very unreliable. Matrix factorization (MF), as an advanced CF technique, plays an important role in recommender systems. MF models [2]–[5] characterize both items and users by vectors in the same space, which are inferred from the observed user-item interactions. Regarding the recommendation as a binary classification problem, logistic regression and its variants [6] are also utilized in recommender systems. However, logistic regression based models are hard to generalize to the feature interactions that never or rarely appear in the training data. Factorization machines [7] model pairwise feature interactions as inner product of latent vectors between features and show promising results. As an extension to FM, Field-aware FM (FFM [8]) enables each feature to have multiple latent vectors to interact with different fields. Recently, deep learning models [9]–[12] are applied to model the complicated feature interactions for recommendation.

在本世纪初，提出了基于内容的过滤[1]，通过考虑项目之间的内容相似性来推荐项目。后来，提出了协同过滤（CF），并对其进行了广泛的研究。 CF的基本原理是，行为相似的用户倾向于偏爱相同的物品，而相似用户所消费的物品往往具有相同的评分。然而，传统的基于CF的方法往往会遭受数据稀缺的困扰，因为从稀疏数据计算出的相似度可能非常不可靠。矩阵分解（MF）作为一种先进的CF技术，在推荐系统中起着重要的作用。 MF模型[2] – [5]通过在相同空间中的向量来表征项目和用户，这些向量是从观察到的用户-项目交互推论得出的。关于推荐作为二元分类问题，逻辑回归及其变体[6]也被用于推荐系统中。然而，基于逻辑回归的模型很难推广到从未或很少出现在训练数据中的特征相互作用。分解机器[7]将成对的特征相互作用建模为特征之间的潜在向量的内积，并显示出令人鼓舞的结果。作为FM的扩展，现场感知FM（FFM [8]）使每个功能都具有多个潜在矢量，可以与不同的字段进行交互。最近，深度学习模型[9] – [12]被用于对复杂的功能交互进行建模以进行推荐。

As a distinguished direction, contextual multi-armed bandits are also utilized to model the interactive nature of recommender systems [13]–[17]. Li et al. apply Thompson Sampling (TS) and Upper Confident Bound (UCB) to balance the tradeoff between exploration and exploitation in [13] and [14], respectively. The authors of [16] propose a dynamic context drift model to address the time varying problem. To integrate the latent vectors of items and users with some exploration, the authors of [15], [17] combine matrix factorization with multi-armed bandits.

作为一个独特的方向，上下文多武装匪徒也被用来对推荐系统的交互性质进行建模[13]-[17]。 Li等。 分别在[13]和[14]中应用汤普森抽样（TS）和上限可信范围（UCB）来平衡勘探与开发之间的权衡。 [16]的作者提出了一种动态上下文漂移模型来解决时变问题。 为了对项目和用户的潜在向量进行整合并进行一些探索，[15]，[17]的作者将矩阵分解与多臂土匪相结合。

However, all these methods suffer from two limitations. First, they consider the recommendation procedure as a static process, i.e., they assume the underlying user’s preference keeps static and they aim to learn the user’s preference as precise as possible. Second, they are learned to maximize the immediate rewards of recommendations, but ignore the longterm benefits that the recommendations can make.

但是，所有这些方法都有两个局限性。 首先，他们将推荐程序视为一个静态过程，即，他们假设基本用户的偏好保持不变，并且旨在尽可能精确地了解用户的偏好。 其次，他们学会了最大程度地提高建议的即时回报，但忽略了建议可以带来的长期利益。

1. RL based Recommendation Techniques

As model-based RL techniques [18], [27] are inapplicable in recommendation scenario due to their high time complexity, most researchers turn to model-free RL techniques. The model-free RL techniques can be divide into two categories: policy-based and value-based

由于基于模型的RL技术[18]，[27]由于时间复杂度高而在推荐方案中不适用，因此大多数研究人员转向无模型RL技术。 无模型的RL技术可以分为两类：基于策略的和基于价值的

Policy-based approaches [28], [32], [33] aim to generate a policy, of which the input is a state, and the output is an action. These works apply deterministic policies, which generates an action directly. Dulac-Arnold et al. [33] resolves the large action space problem by modeling the state in a continuous item embedding space and selecting the items via a neighborhood method. However, as the underlying algorithm is essentially a continuous-action algorithm, its performance may be cursed by the gap between the continuous and discrete action spaces. In [28], [32], the policy network outputs a continuous action representation, and the recommendation is generated by ranking the items with their scores, which are computed by a pre-defined function with the action representation and the item embeddings as input. However, one common limitation of the studies is that they do not carefully learn the state representation.

基于策略的方法[28]，[32]，[33]旨在生成策略，其输入是状态，而输出是动作。 这些工作采用确定性策略，可直接生成操作。 Dulac-Arnold等。 [33]通过对连续项目嵌入空间中的状态进行建模并通过邻域方法选择项目，解决了大型行动空间问题。 但是，由于基础算法本质上是连续动作算法，因此其性能可能会因连续动作空间和离散动作空间之间的差距而降低。 在[28]，[32]中，策略网络输出连续的动作表示，并通过对项目及其得分进行排名来生成推荐，这些得分是由预先定义的函数将动作表示和项目嵌入作为输入来计算的 。 但是，这些研究的一个普遍限制是，他们没有仔细学习状态表示。

For value-based approaches [19], [29], the action with maximum Q-value over all the possible actions is selected as the best action. Zhao et al. [29] take both user’s positive feedback and negative feedback into consideration when modeling user state. Dueling Q-network is utilized in [19], to model Qvalue of a state-action pair. Moreover, a minor update with exploration by dueling bandit gradient descent is proposed. However, such value-based approaches need to evaluate the Q-values of all the actions under a specific state, which is very inefficient when the number of actions is large.

对于基于价值的方法[19]，[29]，将在所有可能的动作中具有最大Q值的动作选择为最佳动作。 赵等。 [29]在对用户状态进行建模时，要同时考虑用户的正面反馈和负面反馈。 在[19]中利用决斗Q网络对状态-动作对的Q值进行建模。 此外，提出了通过对决匪徒梯度下降进行探索的一个小更新。 但是，这种基于值的方法需要评估特定状态下所有动作的Q值，这在动作数量很大时效率很低。

We model the recommendation procedure as a sequential decision making problem, in which the recommender (i.e., agent) interacts with users (i.e., environment) to suggest a list of items sequentially over the timesteps, by maximizing the cumulative rewards of the whole recommendation procedure. More specifically, the recommendation procedure is modeled by an MDP, as follows.

我们将推荐程序建模为一个顺序决策问题，在该问题中，推荐程序（即智能体）与用户（即环境）进行交互，以通过最大化整个推荐程序的累积奖励，在时间步长上顺序地建议项目列表 。 更具体地说，推荐过程由MDP建模，如下所示。

• States S. A state s is the representation of user’s positive interaction history with recommender, as well as her demographic information (if it exists in the datasets).

•状态S。状态s是用户与推荐者的积极互动历史记录以及其人口统计信息（如果数据集中存在）的表示。

• Actions A. An action a is a continuous parameter vector denoted as a ∈ R1×k . Each item it ∈ R1×k 1 has a ranking score, which is defined as the inner product of the action and the item embedding, i.e., ita>. Then the top ranked ones will be recommended.

•动作A。动作a是表示为的连续参数向量。 它的每个项目都有一个排名得分，该得分被定义为动作和项目嵌入的内积，即。 然后将推荐排名最高的。

• Transitions P. The state is modeled as the representation of user’s positive interaction history. Hence, once the user’s feedback is collected, the state transition is determined.

•转换P.状态被建模为用户积极交互历史的表示。因此，一旦收集了用户的反馈，状态转换就确定了。

• Reward R. Given the recommendation based on the action a and the user state s, the user will provide her feedback, i.e., click, not click, or rating, etc. The recommender receives immediate reward R(s, a) according to the user’s feedback.

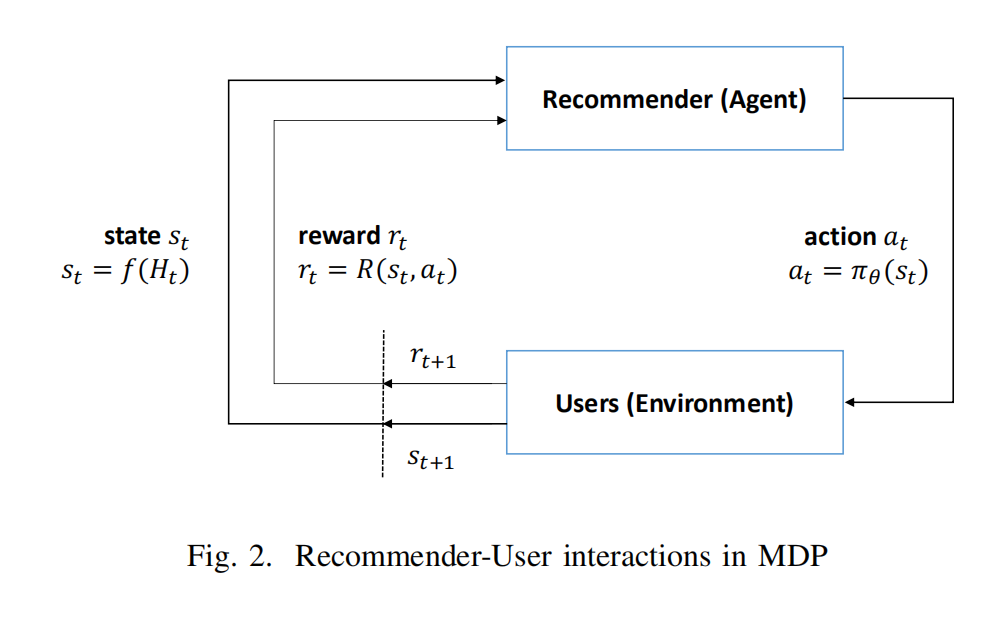
•奖励R。给定基于操作a和用户状态s的推荐，用户将提供她的反馈，即单击，不单击或评级等。推荐人根据以上信息立即获得奖励R（s，a）。

• Discount rate γ. γ ∈ [0, 1] is a factor measuring the present value of long-term rewards. In the case of γ = 0, the recommender considers only immediate rewards but long-term rewards are ignored. On the other hand, when γ = 1, the recommender treats immediate rewards and long-term rewards as equally important.

•折扣因子γ。 γ∈[0，1]是衡量长期奖励现值的因子。 在γ= 0的情况下，推荐者仅考虑立即获得的奖励，而忽略长期获得的奖励。 另一方面，当γ= 1时，推荐者将立即获得的奖励和长期获得的奖励视为同等重要。

Figure 2 illustrates the recommender-user interactions in MDP formulation. Considering the current user state and immediate reward to the previous action, the recommender takes an action. Note that in our model, an action corresponds to neither recommending an item nor recommending a list of items. Instead, an action is a continuous parameter vector. Taking such an action, the parameter vector is used to determine the ranking scores of all the candidate items, by performing inner product with item embeddings. All the candidate items are ranked according to the computed scores and Top-N items are recommended to the user. Taking the recommendation from the recommender, the user provides her feedback to the recommender and the user state is updated accordingly. The recommender receives rewards according to the user’s feedback. Without loss of generalization, a recommendation procedure is a T timestep2 trajectory as (s0, a0, r0, s1, a1, r1, ..., sT T1, aT T1, rT T1, sT ).

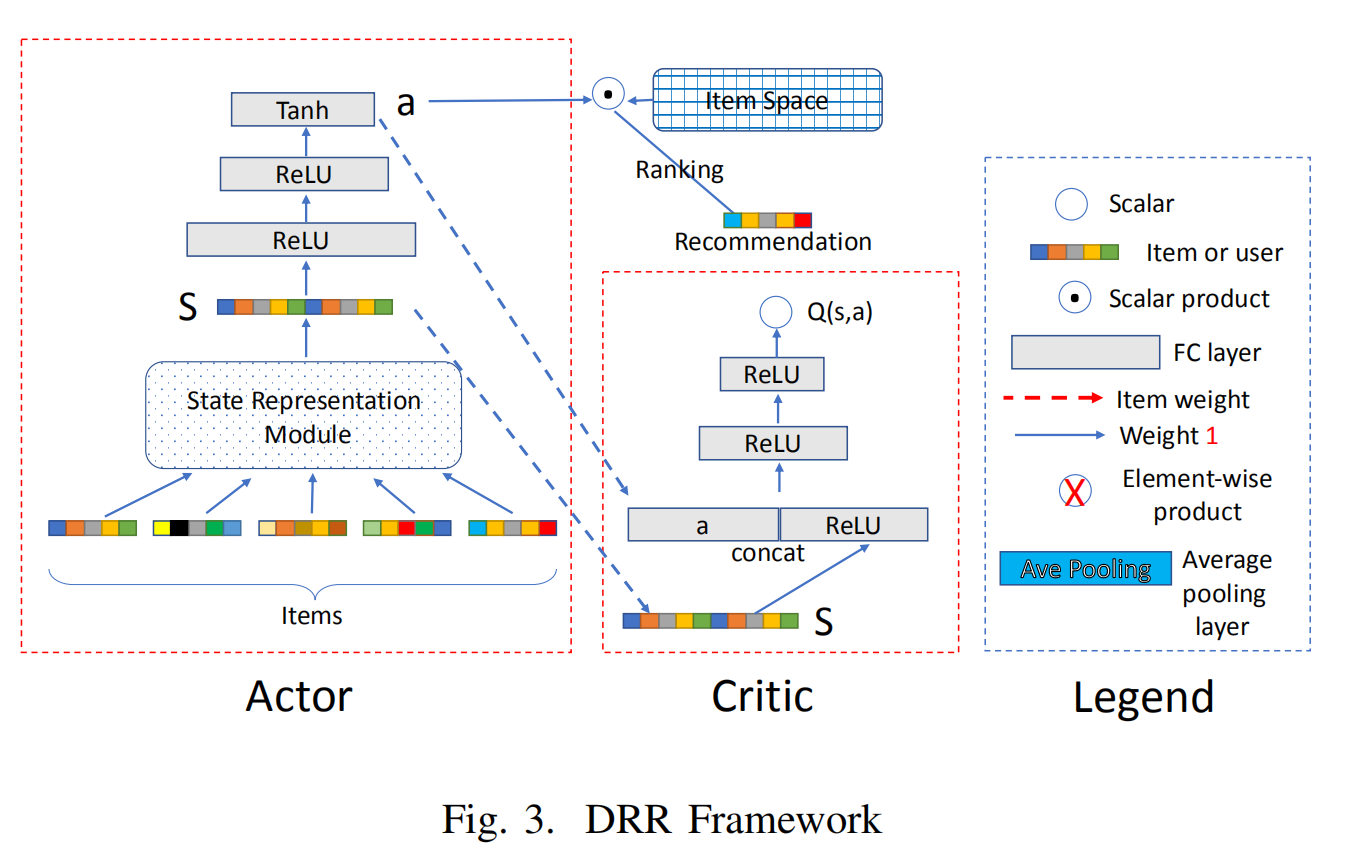
在每个时间步，推荐代理根据用户状态 采取动作 *，*并获得奖励。 因此，用户状态随着转换 更新为 。 推荐代理的目标是找到一个最优策略 ，最大化推荐系统的预期累积奖励。

图2说明了MDP制定中的推荐者与用户的交互。 考虑到当前用户状态和对上一个操作的立即奖励，推荐器将执行一个操作。 请注意，在我们的模型中，动作既不建议项也不建议项列表。 相反，一个动作是一个连续的参数向量。 采取这样的动作，通过执行带有项目嵌入的内积，参数向量用于确定所有候选项目的排名得分。 根据计算出的分数对所有候选项目进行排名，并向用户推荐排名前N的项目。 用户从推荐者那里获得推荐，用户将其反馈信息提供给推荐者，并相应地更新用户状态。 推荐人根据用户的反馈获得奖励。 不失一般性，推荐过程是T timestep2轨迹为（s0，a0，r0，s1，a1，r1，...，sT T1，aT T1，rT T1，sT）。

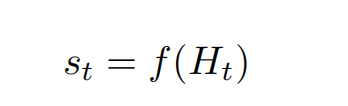
IV. THE PROPOSED DRR FRAMEWORK IV.所提出的DRR框架

As aforementioned in Section 1, conventional recommendation techniques suffer from either a lack of sequential modeling or ignoring the long-term rewards, or both. To address the drawbacks, we propose a deep reinforcement learning based recommendation framework (DRR) based on the ActorCritic learning scheme. Also, different from some recent RL studies, we carefully and explicitly build a state representation module to model the interactions between the users and items. Next, we will first elaborate the Actor network, Critic network and the state representation module respectively, which are essentially the three key ingredients in our framework; then the training and evaluation procedures will be presented to show how to learn and use the DRR framework.

如第1节所述，传统的推荐技术要么缺少顺序建模，要么忽略了长期奖励，或者两者兼而有之。 为了解决这些缺点，我们提出了一个基于ActorCritic学习方案的基于深度强化学习的推荐框架（DRR）。 另外，与最近的一些RL研究不同，我们仔细而明确地构建了一个状态表示模块，以对用户和项目之间的交互进行建模。 接下来，我们将首先分别详细说明Actor网络，Critic网络和状态表示模块，它们实际上是我们框架中的三个关键要素。 然后将介绍训练和评估程序，以显示如何学习和使用DRR框架。



1. *Three Key Ingredients in DRR*
2. The Actor network: The Actor network, also called the policy network, is depicted on the left part of Figure 3. For a given user, the network accounts for generating an action a based on her state s. Let us explain the network from the input to the output part. In DRR, the user state, denoted by the embeddings of her n latest positively interacted items, is regarded as the input. Then the embeddings are fed into a state representation module (which will be introduced in details later) to produce a summarized representation s for the user. For instance, at timestep t, the state can be defined in Eq. (1):
3. Actor网络：Actor网络，也称为策略网络，显示在图3的左侧。对于给定的用户，该网络负责根据其状态s生成动作a。 让我们解释一下从输入到输出部分的网络。 在DRR中，将用户状态（由她的n个最新的积极互动的项目的嵌入表示）视为输入。 然后将嵌入内容馈送到状态表示模块（稍后将对其进行详细介绍）以为用户生成汇总表示。 例如，在时间步t，可以在等式中定义状态。 （1）：



where f(·) stands for the state representation module, Ht = {i1, ..., in} denotes the embeddings of the latest positive interaction history, and it ∈ R1×k is a k-dimensional vector. When the recommender agent recommends an item it, if the user provides positive feedback, then in the next timestep, the state is updated to st+1 = f(Ht+1), where Ht+1 = {i2, ..., in, it}; otherwise, Ht+1 = Ht. The reasons to define the state in such a manner are two folds: (i) a superior recommender system should cater to the users’ taste, i.e., what items the users like; (ii) the latest records represent the users’ recent interests more precisely.

其中代表状态表示模块，表示最新的正向交互历史的嵌入，并且是k维向量。当推荐代理推荐它时，如果用户提供了肯定的反馈，则在下一个时间步中，状态将更新为，其中; 否则，。 以这种方式定义状态的原因有两个方面：

优秀的推荐系统应满足用户的口味，即用户喜欢什么项目；

最新记录更准确地代表了用户的近期兴趣。

Actor 网络，也称为策略网络，如图 1 左上角所示。 对于给定的用户，网络会根据用户的状态 s 生成动作 a。用户的状态由状态表示模块建模和更新。例如，在时间步 t，状态可以定义为：，其中 代表状态表示模块， 表示最近的积极交互历史， 是一个 维向量，表示第 项的嵌入。当代理推荐一个项目 时，如果用户提供了正反馈，那么在下一个时间步，状态更新为 ，其中否则， 。通过两个 ReLU 层和一个Tanh 层，状态表示 s 被转换为动作 作为 Actor 网络的输出。特别地，动作 a 被定义为由连续参数向量 表示的排序函数。根据a，项目v∈V的排序分数定义为

Finally, by two ReLU layers and one Tanh layer, the state representation s is transformed into an action as the output of the Actor network. Particularly, the action a is defined as a ranking function represented by a continuous parameter vector . By using the action, the ranking score of the item it is defined as:

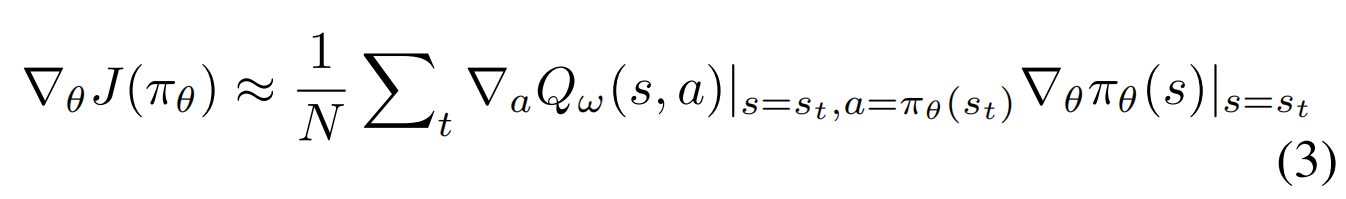
最后，通过两个ReLU层和一个Tanh层，将状态表示s转换为动作作为Actor网络的输出。 特别地，将动作a定义为由连续参数向量表示的排序函数。通过使用该动作，该项目集的排名得分定义为：

Then, the top ranked item (w.r.t. the ranking scores) is recommended to the user. Note that, the widely used ε-greedy exploration technique is adopted here.

然后，将排名最高的项目（w.r.t.排名分数）推荐给用户。 注意，这里采用了广泛使用的ε-greedy探索技术。

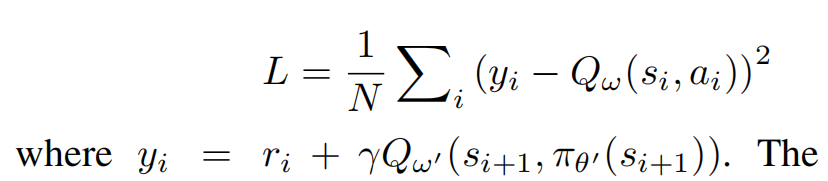
1. The Critic network: The Critic part in DRR, shown as the middle part of Figure 3, is a Deep Q-Network [21], which leverages a deep neural network parameterized as Qω(s, a) to approximate the true state-action value function Qπ(s, a), namely, the Q-value function. The Q-value function reflects the merits of the action policy generated by the Actor network. Specifically, the input of the Critic network is the user state s generated by the user state representation module and the action a generated by the policy network, and the output is the Q-value, which is a scalar. According to the Q-value, the parameters of the Actor network are updated in the direction of improving the performance of action a, i.e., boosting Qω(s, a). Based on the deterministic policy gradient theorem [34], we can update the Actor by the sampled policy gradient shown in Eq.(3):
2. The Critic network：DRR中的Critic部分，如图3的中间部分所示，是一个深度Q网络[21]，它利用参数化为的深度神经网络来逼近真实状态行为值函数，即Q-value函数。Q-value函数反映了Actor网络生成的动作策略的优点。具体来说，Critic网络的输入是用户状态表示模块生成的用户状态s和策略网络生成的动作a，输出是标量q值。根据q值，对Actor网络的参数进行更新。基于确定性策略梯度定理[34]，我们可以根据Eq.(3)所示的采样策略梯度对Actor进行更新:

DRR 中的 Critic 网络（也称为价值网络），如图 1 中间部分所示，是一个深度 Q 网络 [18]，它利用参数化为 的深度神经网络来逼近真实的状态动作值函数，即Q-value函数。 Q-value反映了 Actor 网络生成的策略的优劣。 具体来说，Critic网络的输入是用户状态表示模块生成的用户状态s和策略网络生成的动作a，输出是Q值，是一个标量。 根据Q值，Actor网络的参数朝着提高动作a性能的方向更新，即提升。 基于确定性策略梯度定理 [45]，我们可以通过采样的策略梯度更新 Actor



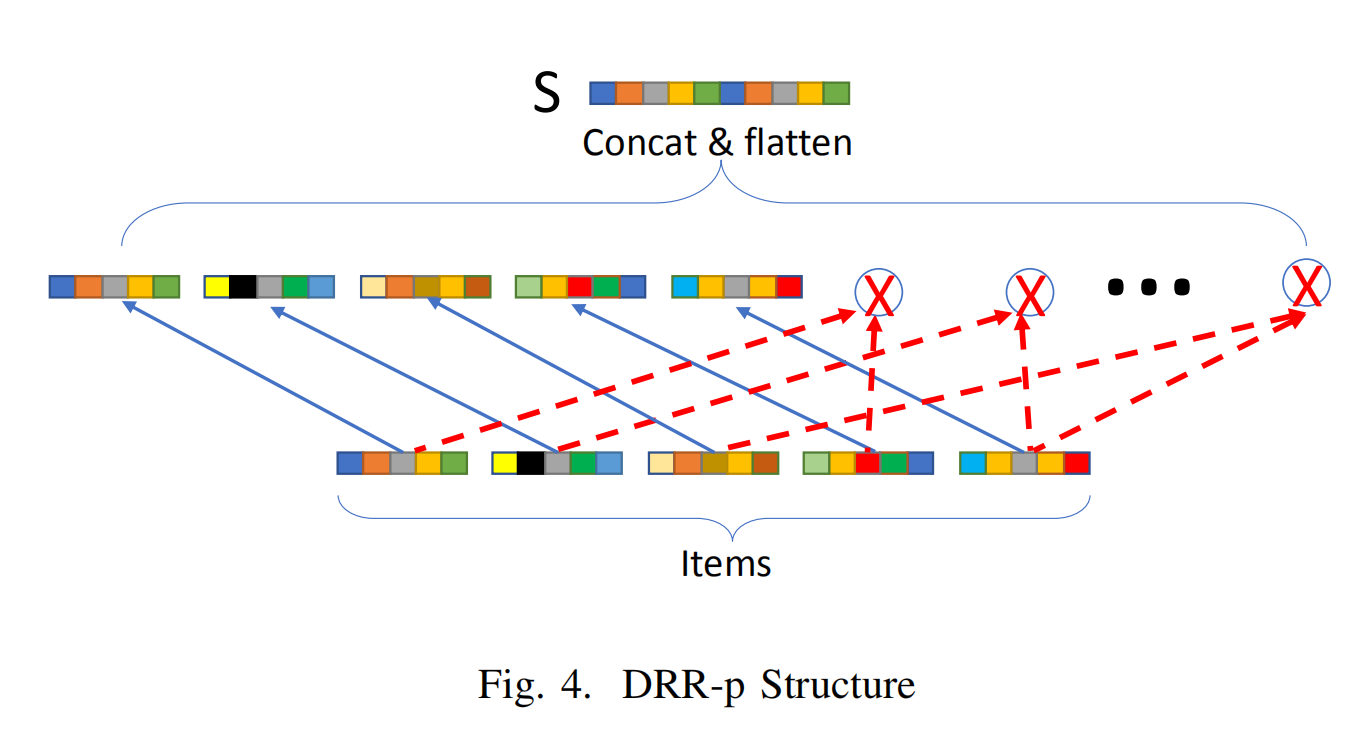
where J(πθ) is the expectation of all possible Q-values that follow the policy πθ. Here the mini-batch strategy is utilized and N denotes the batch size. Moreover, the Critic network is updated accordingly by the temporal-difference learning approach [20], i.e., minimizing the mean squared error shown in Eq.(4):

式中，是遵循策略的所有可能q值的期望。这里采用小批量策略，N为批量大小。使用时序差分学习方法[20]对批评网络进行相应更新，即最小化式(4)所示的均方误差:



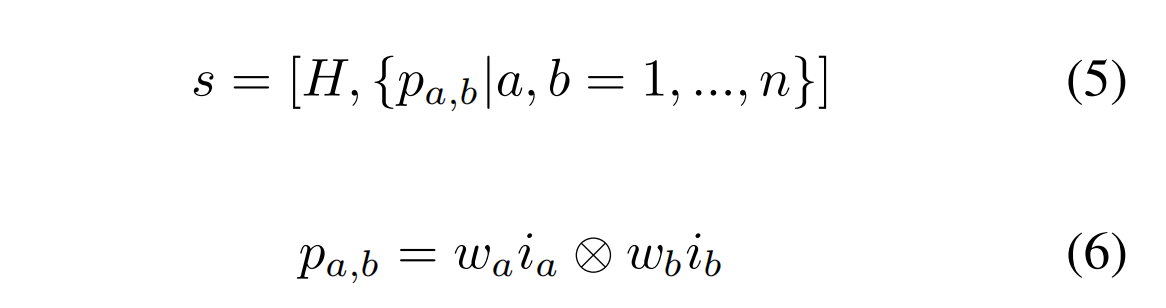
The target network [35] technique is also adopted in DRR framework, where ω0 and θ0 is the parameters of the target Critic and Actor network.

1. The State Representation Module: As noted above, the state representation module plays an important role in both the Actor network and Critic network. Hence, it is very crucial to design a good structure to model the state. In [10], [11], it has been shown that modeling the feature interactions explicitly can boost the performance of a recommendation system. Inspired by the studies, we propose to design the state representation module by explicitly modeling the interactions between the users and items. Specifically, we develop three structures, which will be elaborated next.
2. 状态表示模块：如上所述，状态表示模块在Actor网络和Critic网络中均扮演着重要角色。 因此，设计一个好的状态模型非常关键。 在[10]，[11]中，已经表明对特征交互进行建模可以显着提高推荐系统的性能。 受研究的启发，我们建议通过显式地建模用户与项目之间的交互来设计状态表示模块。 具体来说，我们开发了三种结构，接下来将详细说明。



• DRR-p. Inspired by [10], [11], we propose a product based neural network for the state representation module,which is depicted in Figure 43 . The structure is named as DRR-p, which utilizes a product operator to capture the pairwise local dependency between items. We can see that the structure clones the representations of the n items from H = {i1, ..., in}. In addition, it computes the pairwise interactions between the n items, by using the element-wise product operator. As a result, n(n n 1)/2 new features vectors are yielded, which will be concatenated with the cloned vectors as the state representation. We note that in the element-wise product part, a weight is also learned for each item to show its importance. Hence, in DRR-p the state representation module can be formally stated as follows:

*,*

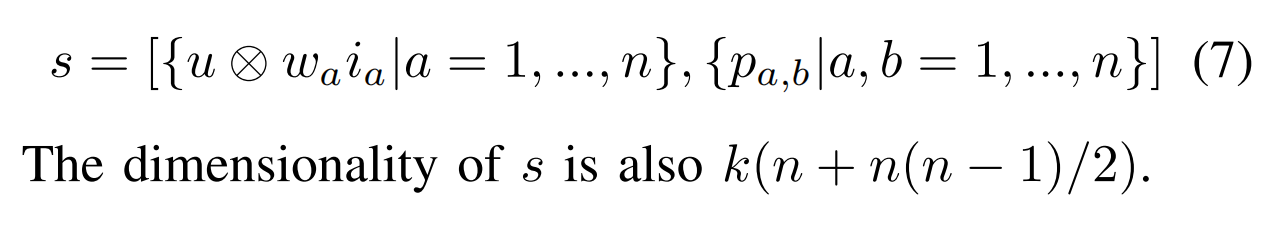
•DRR-p。 受[10]，[11]的启发，我们为状态表示模块提出了基于项目的神经网络，如图43所示。 该结构称为DRR-p，它利用乘积运算符捕获项之间的成对局部依赖关系。 我们可以看到该结构从中复制了n个项的表示。 此外，它通过使用逐元素乘积运算符来计算n个项目之间的成对交互。 结果，产生了个新特征向量，这些向量将与克隆的向量连接起来作为状态表示。 我们注意到，在元素级产品部分中，还为每个项目学习了一个权重，以显示其重要性。 因此，在DRR-p中，状态表示模块可以正式表示为：

where ⊗ denotes the element-wise product, wa is a scalar indicating the importance of item ia, and pa,b is a k-dimensional vector which models the interactions between item ia and ib. The dimensionality of s is k(n + n(n n 1)/2).

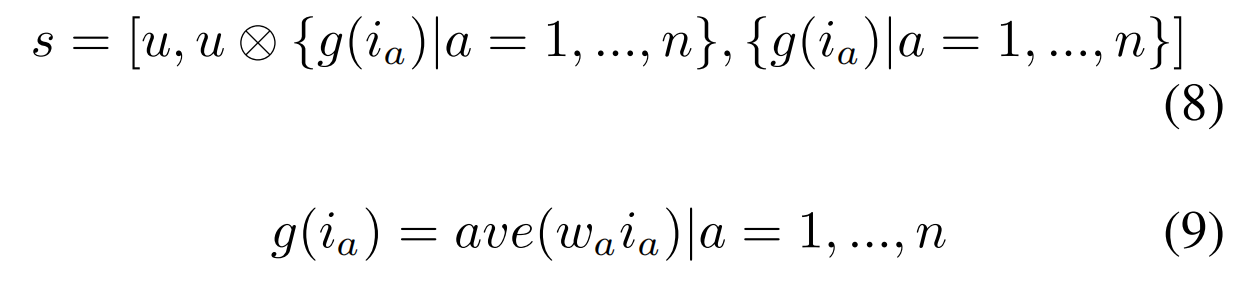
其中⊗为元素的乘积，为一个标量，表示项的重要性，为k维向量，对项和项之间的相互作用进行建模，的维数为k(n+n(n-1)/2)。

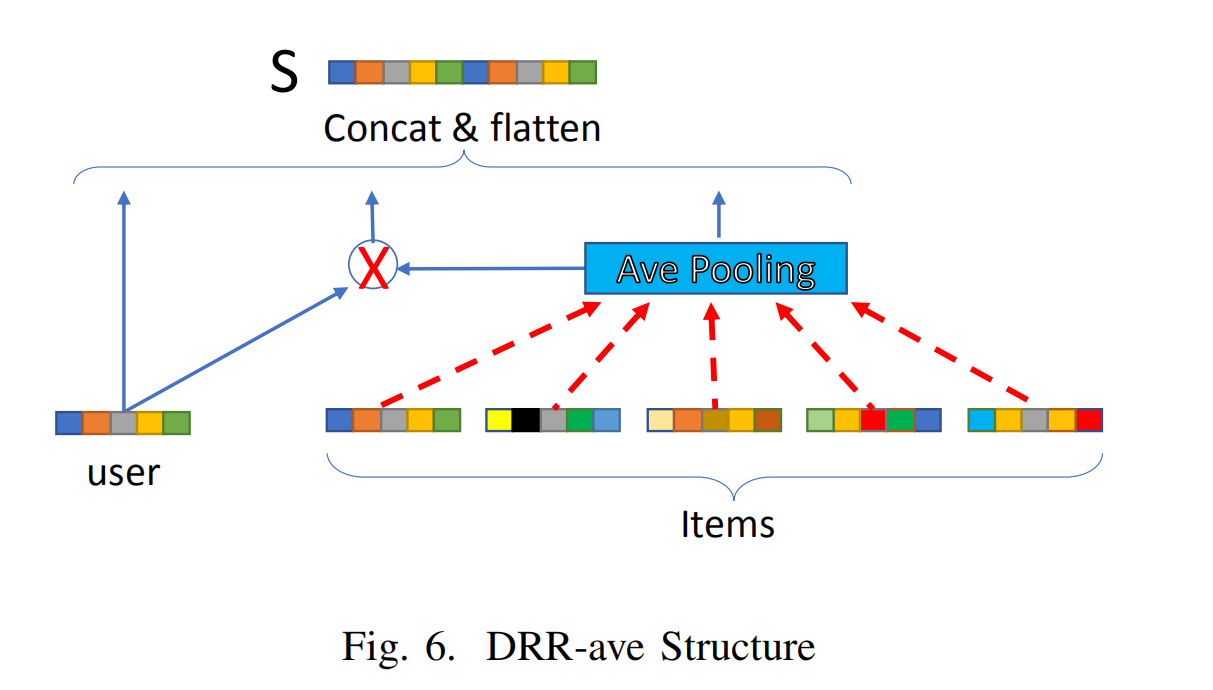
• DRR-u. Though DRR-p can model the pairwise local dependency between items, the user-item interactions are neglected. To remedy this, we design another structure in Figure 5, which is referred as DRR-u. In DRR-u, we can see that the user embedding is also incorporated. In addition to the local dependency between items, the pairwise interactions of user-item are also taken into account. Formally, the state representation module can be expressed as:

•DRR-u。尽管DRR-p可以对项目之间的局部依赖关系进行建模，但忽略了用户与项目之间的交互。为了解决这个问题，我们在图5中设计了另一个结构，称为DRR-u。在DRR-u中，我们可以看到也加入了用户嵌入。除了项之间的局部依赖性，user-item的成对交互也被考虑在内。在形式上，状态表示模块可以表示为:



• DRR-ave. In DRR-p and DRR-u structures, the interactions between users and items can be exploited and modeled. For the two structures, it is not difficult to find that the positions of items in H matters, e.g., the state representations of H1 = {ia, ib, ic} and H2 = {ic, ib, ia} are different. When H is large, we expect the positions of items really matter, because H denotes a long-term sequence; whereas memorizing the positions of items may lead to overfitting if the sequence H is a short-term one. Hence, we design another structure by eliminating the position effects, which is depicted in Figure 6. As an average pooling layer is adopted, we call the structure DRR-ave. We can see from Figure 6 that the embeddings of items in H are first transformed by a weighted average pooling layer. Then, the resulting vector is leveraged to model the interactions with the input user. Finally, the embedding of the user, the interaction vector, and the average pooling result of items are concatenate into a vector to denote the state representation. Formally, the DRR-ave structure can be expressed as:

•DRR-ave。在DRR-p和DRR-u结构中，用户和项目之间的交互可以被利用和建模。对于这两种结构，不难发现H物中元素的位置，如和的状态表示是不同的。当H大时，由于H表示一个长期序列，因此我们预期项目的位置确实很重要。 而如果序列H是短期序列，则记住项目的位置可能会导致过度拟合。 因此，我们通过消除位置效应来设计另一种结构，如图6所示。当采用平均池化层时，我们将其称为DRR-ave结构。 从图6中我们可以看到，H中的项嵌入首先是通过加权平均池化层进行转换的。 然后，利用所得的向量来建模与输入用户的交互。 最后，将用户的嵌入，交互向量和项的平均合并结果连接到一个向量中，以表示状态表示。 形式上，DRR-ave结构可以表示为：



Here g(·) indicates the weighted average pooling layer. The dimensionality of s in DRR-ave is 3k.

此处的g（·）表示加权平均池化层。 在DRR-ave中s的维数为3k。

1. *Training Procedure of the DRR Framework*

Next, we introduce how to train the DRR framework. We first present the overall idea and then discuss the detailed algorithm. As aforementioned, DRR utilizes the users’ interaction history with the recommender agent as training data. During the procedure, the recommender takes an action at following the current recommendation policy πθ(st) after observing the user (environment) state st, then it obtains the feedback (reward) rt from the user, and the user state is updated to st+1. According to the feedback, the recommender updates its recommendation policy. In this work, we utilize deep deterministic policy gradient (DDPG) [35] algorithm to train the proposed DRR framework, as detailed in Algorithm 1.

接下来，我们介绍如何训练DRR框架。 我们首先提出总体思路，然后讨论详细的算法。 如前所述，DRR利用用户与推荐人代理的交互历史作为培训数据。 在该过程中，推荐者在观察到用户（环境）状态之后，遵循当前推荐策略措施，然后从用户那里获得反馈（奖励），并将用户状态更新为。 根据反馈，推荐者更新其推荐策略。 在这项工作中，我们利用深度确定性策略梯度（DDPG）[35]算法训练提出的DRR框架，如算法1所述。

Specifically, in timestep t, the training procedure mainly includes two phases, i.e., transition generation (lines 7-12) and model updating (lines 13-17). For the first stage, the recommender observes the current state st that is calculated by the proposed state representation module, then generates an action at = πθ(st) according to the current policy πθ with ε-greedy exploration, and recommends an item it according to the action at by Eq. (2) (lines 8-9). Subsequently, the reward rt can be calculated based on the feedback of the user to the recommended item it, and the user state is updated (lines 10-11). Finally, the recommender agent stores the transition (st, at, rt, st+1) into the replay buffer D (line 12).

具体地，在时间步t中，训练过程主要包括两个阶段，即转换（transition）生成（第7-12行）和模型更新（第13-17行）。 对于第一阶段，推荐者观察由state representation module计算出的当前状态，根据当前策略使用ε-greedy探索获得，并根据推荐当前最优项目（第8-9行）。 随后，可以基于用户对推荐项目的反馈来计算奖励，并且更新用户状态（第10-11行）。 最后，推荐者将转换存储到重播缓冲区D中（第12行）。

In the second stage, the model updating, the recommender samples a minibatch of N transitions with widely used prioritized experience replay [36] sampling technique (line 13), which is essentially an importance sampling strategy. Then, the recommender updates the parameters of the Actor network and Critic network according to Eq. (3) and Eq. (4) respectively (line 14-16). Finally, the recommender updates the target networks’ parameters with the soft replace strategy.

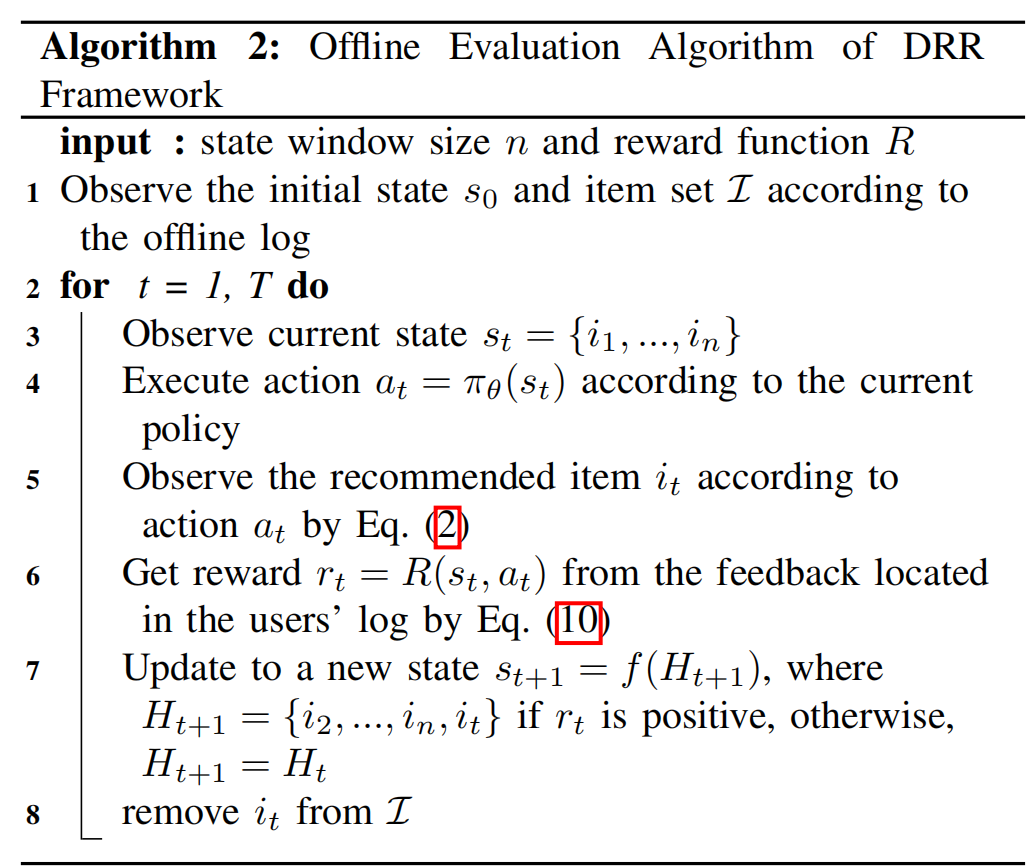
在第二阶段，模型更新阶段，智能体对小批量的N个转换进行采样，采用了广泛使用的优先体验重放[36]采样技术(第13行)，这本质上是一种重要性采样策略。然后，推荐器根据式(3)和式(4)分别更新Actor网络和critical网络的参数(第14-16行)。最后，用软替换策略更新目标网络的参数。

1. *Evaluation*

In this subsection, we discuss how to evaluate the models with a environment simulator. The most straightforward way to evaluate the RL based models is to conduct online experiments on recommender systems where the recommender directly interacts with users. However, the underlying commercial risk and the costly deployment on the platform make it impractical. Therefore, throughout the testing phase, we conduct the evaluation of the proposed models on public offline datasets and propose two ways to evaluate the models, which are the offline evaluation and the online evaluation.

在本小节中，我们将讨论如何使用环境模拟器评估模型。评估基于RL的模型最直接的方法是在推荐系统上进行在线实验，让推荐者直接与用户交互。然而，潜在的商业风险和在平台上部署的昂贵成本使其不切实际。因此，在整个测试阶段，我们在公共离线数据集上对所提出的模型进行评估，并提出两种评估模型的方式，即离线评估和在线评估。

1. Offline evaluation: Intuitively, the offline evaluation of the trained models is to test the recommendation performance with the learned policy, which is described in Algorithm 2. Specifically, for a given session Sj , the recommender only recommends the items that appear in this session, denoted as I(Sj ), rather than the ones in the whole item space. The reason is that we only have the ground truth feedback for the items in the session in the recoreded offline log. For each timestep, the recommender agent takes an action at according to the learned policy πθ, and recommends an item it ∈ I(Sj ) based on the action at by Eq. (2) (lines 4-5). After that, the recommender observes the reward rt = R(st, at) according to the feedback of the recommended item it by Eq. (10) (lines 5-6). Then the user state is updated to st+1 and the recommended item it is removed from the candidate set I(Sj ) (lines 7-8). The offline evaluation procedure can be treated as a rerank procedure of the candidate set by iteratively selecting an item w.r.t. the action generated by the Actor network in DRR framework. Moreover, the model parameters are not updated in the offline evaluation.
2. 离线评估:从直观上看，训练后的模型的离线评估是用学习后的策略来测试推荐的性能，在算法2中描述。具体地说，对于给定的session ，推荐器只推荐出现在这个session中的项目(表示为)，而不是整个项目空间中的项目。原因是我们在记录的离线日志中只有会话中的项目的ground truth反馈。对于每个时间步，推荐agent根据学习到的策略采取一个动作，根据动作，通过Eq.(2)(第4-5行)推荐一个的项目。然后，根据Eq.(10)(第5-6行)对被推荐项目的反馈，被推荐者观察奖励。然后，用户状态被更新为，并从候选集中删除推荐的项目(第7-8行)。离线评估过程可以看作是对候选集的重新排序过程，通过迭代地选择一个项目。离线评估时，模型参数没有更新。



1. Online evaluation with environment simulator: As aforementioned that it is risky and costly to directly deploy the RL based models on recommender systems. Therefore, we conduct online evaluation with an environment simulator. In this paper, we pretrain a PMF [37] model as the environment simulator, i.e., to predict an item’s feedback that the user never rates before. The online evaluation procedure follows Algorithm 1, i.e., the parameters continuously update during the online evaluation stage. Its major difference from Algorithm 1 is that the feedback of a recommended item is observed by the environment simulator. Moreover, before each recommendation session starting in the simulated online evaluation, we reset the parameters back to θ and ω which is the policy learned in the training stage for a fair comparison.

2）使用环境模拟器进行在线评估：如前所述，将基于RL的模型直接部署在推荐系统上既冒险又昂贵。 因此，我们使用环境模拟器进行在线评估。 在本文中，我们预训练了PMF [37]模型作为环境模拟器，即预测用户从未评估过的项目反馈。 在线评估程序遵循算法1，即参数在在线评估阶段不断更新。 它与算法1的主要区别在于，环境模拟器可以观察推荐项目的反馈。 此外，在每次推荐会议开始于模拟在线评估之前，我们将参数重置为θ和ω，这是在培训阶段学到的策略，用于公平比较。

V. EXPERIMENT

A. Datasets and Evaluation Metrics We adopt the following publicly available datasets from the real world to conduct the experiments:

• MovieLens (100k)4 . A benchmark dataset comprises of 0.1 million ratings from users to the recommended movies on MovieLens website.

• Yahoo! Music (R3)5 . This dataset contains over 0.36 million ratings of songs collected from two different sources. The first source consists of ratings provided by users during normal interactions with Yahoo! Music services. The second source consists of ratings of randomly selected songs collected during an online survey by Yahoo! Research. We normalize the ratings to discrete values from 1 to 5.

• MovieLens (1M)6 . A benchmark dataset includes of 1 million ratings from the MovieLens website.

• Jester (2)7 . This dataset contains over 1.7 million realvalue ratings (-10.0 to +10.0) over jokes in an online joke recommender system.

A.数据集和评估指标我们采用来自现实世界的以下公共可用数据集来进行实验：

•MovieLens(100k)4。 基准数据集包括用户对MovieLens网站上推荐电影的10万评分。

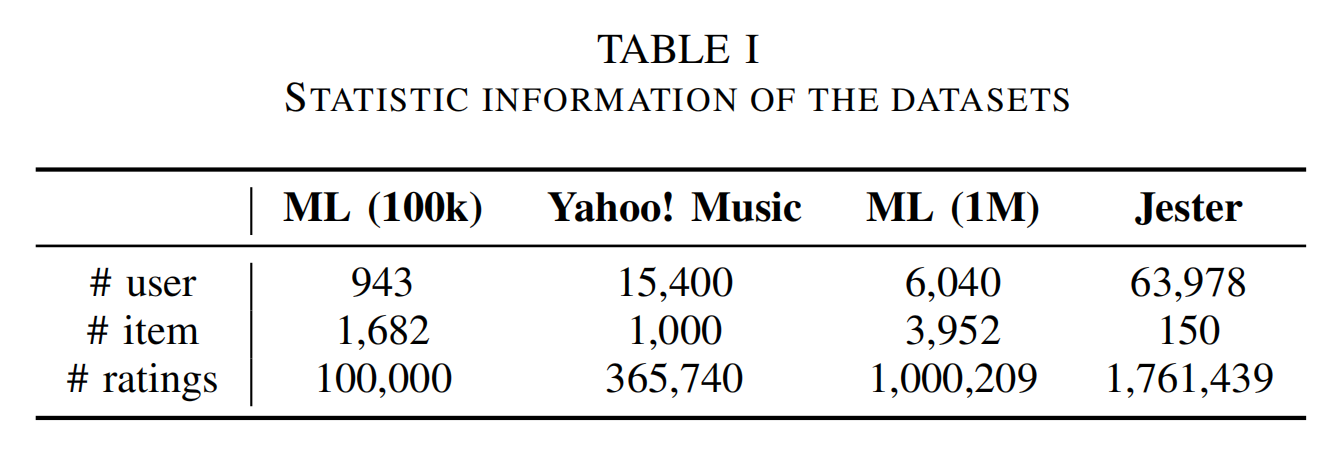
•Yahoo! Music(R3)5。。该数据集包含从两个不同来源收集的超过36万首歌曲评分。第一个来源包括用户在与雅虎正常互动期间提供的评分!第二个数据来源是在雅虎在线调查中随机挑选的歌曲的评分。这里我们将评分标准化为从1到5的离散值。

•MovieLens（1M）6。 基准数据集包含MovieLens网站上的100万个评分。

•Jester (2)7。 该数据集包含在线笑话推荐系统中超过170万个笑话的真实价值评分（-10.0至+10.0）。

Note that except for Jester, the ratings in the other datasets are discrete values from 1 to 5, and the statistic information of the datasets is given in Table I. The MovieLens (100k) and MovieLens (1M) are abbreviated as ML (100k) and ML (1M) respectively.

需要注意的是，除Jester外，其他数据集的评分均为1 - 5的离散值，数据集的统计信息见表i。MovieLens (100k)和MovieLens (1M)分别简写为ML (100k)和ML (1M)。



We conduct both offline and simulated online evaluation on these four datasets. For the offline evaluation, we utilize Precision@k and NDCG@k as the metrics to measure the performance of the proposed models. For the simulated online evaluation, we leverage the total accumulated rewards as the metric.

我们对这四个数据集进行离线和模拟在线评估。

对于离线评估，我们利用Precision @ k和NDCG @ k作为衡量提议模型性能的指标。

对于模拟的在线评估，我们利用累积的总奖励作为指标。

b.Compared Methods

We compare the proposed methods with some representative baseline methods. For the offline evaluation, we compare to conventional methods including Popularity, PMF [37] and SVD++ [38], and a RL based method DRR-n. Moreover, the online evaluation baselines contain the state-of-the-art multiarmed bandits methods LinUCB [39] and HLinUCB [40] and the DRR-n as well.

我们将提出的方法与一些代表性的基线方法进行了比较。 对于离线评估，我们将其与包括“流行度”，PMF [37]和SVD ++ [38]在内的常规方法以及基于RL的DRR-n方法进行了比较。 此外，在线评估基准还包含最先进的多臂土匪方法LinUCB [39]和HLinUCB [40]以及DRR-n。

• **Popularity** recommends the most popular item, i.e., the item with the highest average rating or the items with largest number of positive ratings from current available items to the users at each timestep. 在每个时间步向用户推荐最受欢迎的商品，即平均评分最高的商品或正面评价数量最多的商品。

• **PMF** makes a matrix decomposition as SVD, while it only takes into account the non zero elements. 像SVD一样进行矩阵分解，而只考虑非零元素。

• **SVD++** mixes strengths of the latent model as well as the neighborhood model. 混合了潜在模型和邻域模型的优势。

• **LinUCB** selects an arm (item) according to the estimated upper confidence bound of the potential reward. 根据潜在奖励的置信度上限选择一个手臂（项目）。

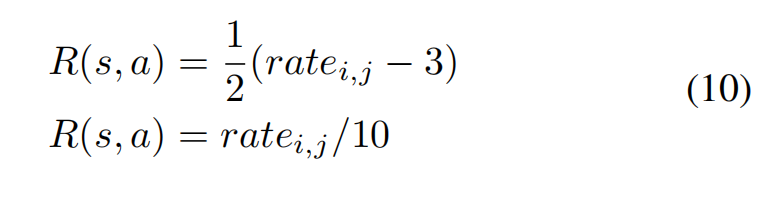
• **HLinUCB** further learns hidden features for each arm to model the potential reward. 进一步学习每条手臂的隐藏特征，以模拟潜在的奖励。

• **DRR-n** simply utilizes the concatenation of the item embeddings to represent user state, which is widely used in previous studies. Although it is under the DRR framework, we treat this method as a baseline to assess the effectiveness of our proposed state representation module. 简单地利用项目嵌入的串联来表示用户状态，这在先前的研究中已广泛使用。 尽管它在DRR框架下，但我们将此方法作为评估我们建议的状态表示模块有效性的基准。

1. *Experimental Settings*

For each dataset, we choose 80% of the interactions in each user session as the training set, and leave the rest as the testing set. Moreover, for MovieLens (100k), Yahoo! Music and MovieLens (1M), the positive ratings are 4 and 5, while for Jester, the positive ones are those higher than 0. The number of latest positively rated items n, which is empirically set to 5. We perform PMF to pretrain the 100-dimensional embeddings of the users and items. Moreover, in each episode, we do not recommend repeated items, i.e., we remove the ones already recommended from the candidate set. The discount rate γ is 0.9. We utilize Adam optimizer for all the RL based methods with L2-norm regularization to prevent overfitting. As for the reward function, we empirically normalize the ratings into range [-1 ,1] and utilize the normalized ones as the feedback of the corresponding recommendations. For instance, in timestep t, the recommender agent recommends an item j to user i, (denoted as action a in state s), and the rating ratei,j comes from the interaction logs if user i actually rates item j, or from a predicted value by the simulator otherwise. Therefore, the reward function can be defined as follows:

对于每个数据集，我们将每个用户会话中80％的互动选择为训练集，其余的作为测试集。此外，对于MovieLens（100k），Yahoo！音乐和MovieLens（1M）的正面评分为4和5，而对于Jester，正面评分为大于0的那些。最新正面评分项目的数量n（根据经验设置为5）。我们执行PMF来预训练用户和项目的100维嵌入。此外，在每个幕中，我们都不推荐重复的项目，即，我们从候选集中删除已经推荐的项目。贴现率γ为0.9。我们将Adam优化器用于所有具有L2范数正则化的基于RL的方法，以防止过度拟合。至于奖励函数，我们根据经验将等级归一化为范围[-1，1]，并利用归一化的等级作为相应建议的反馈。例如，在时间步t中，推荐代理将项目j推荐给用户i（表示为状态s中的动作a），如果用户i实际对项目j进行评级，则评级比率i，j来自交互日志，或者来自否则由仿真器预测值。因此，奖励函数可以定义如下：



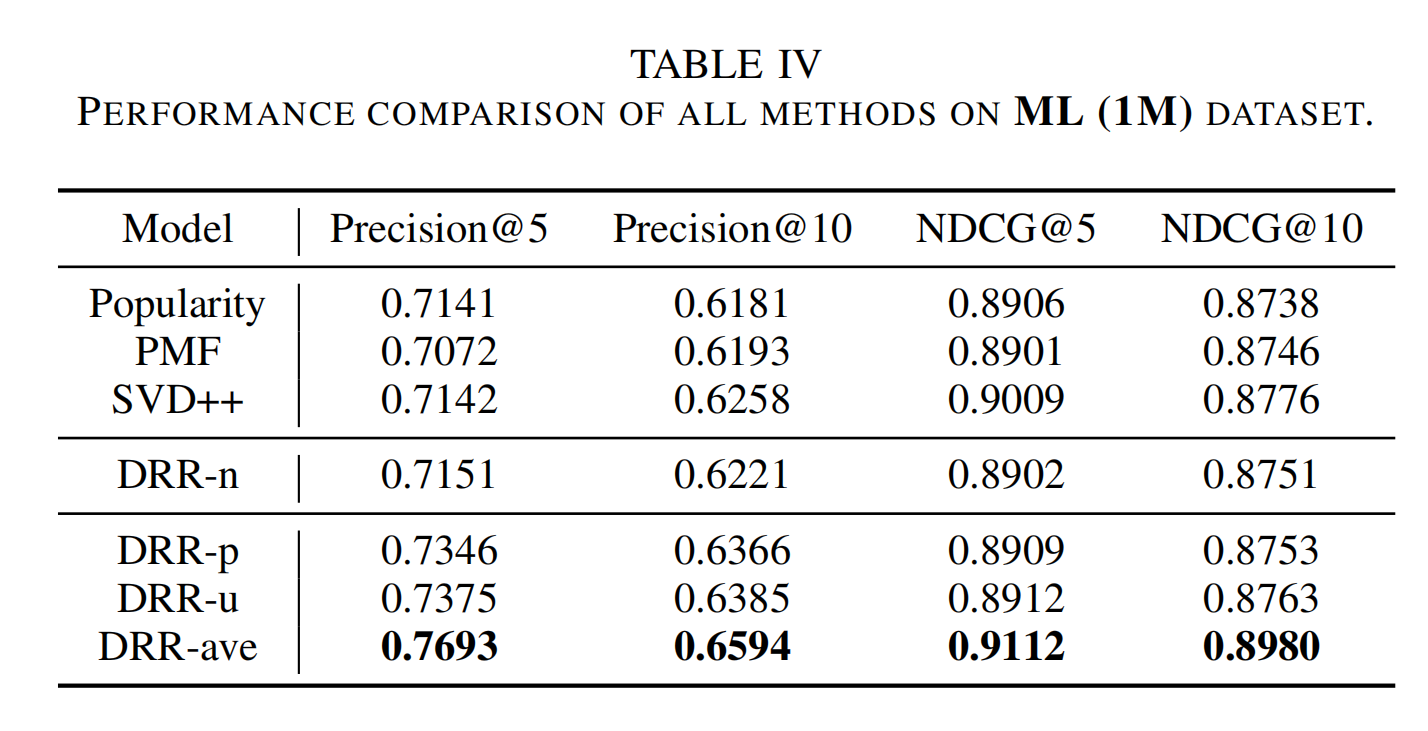
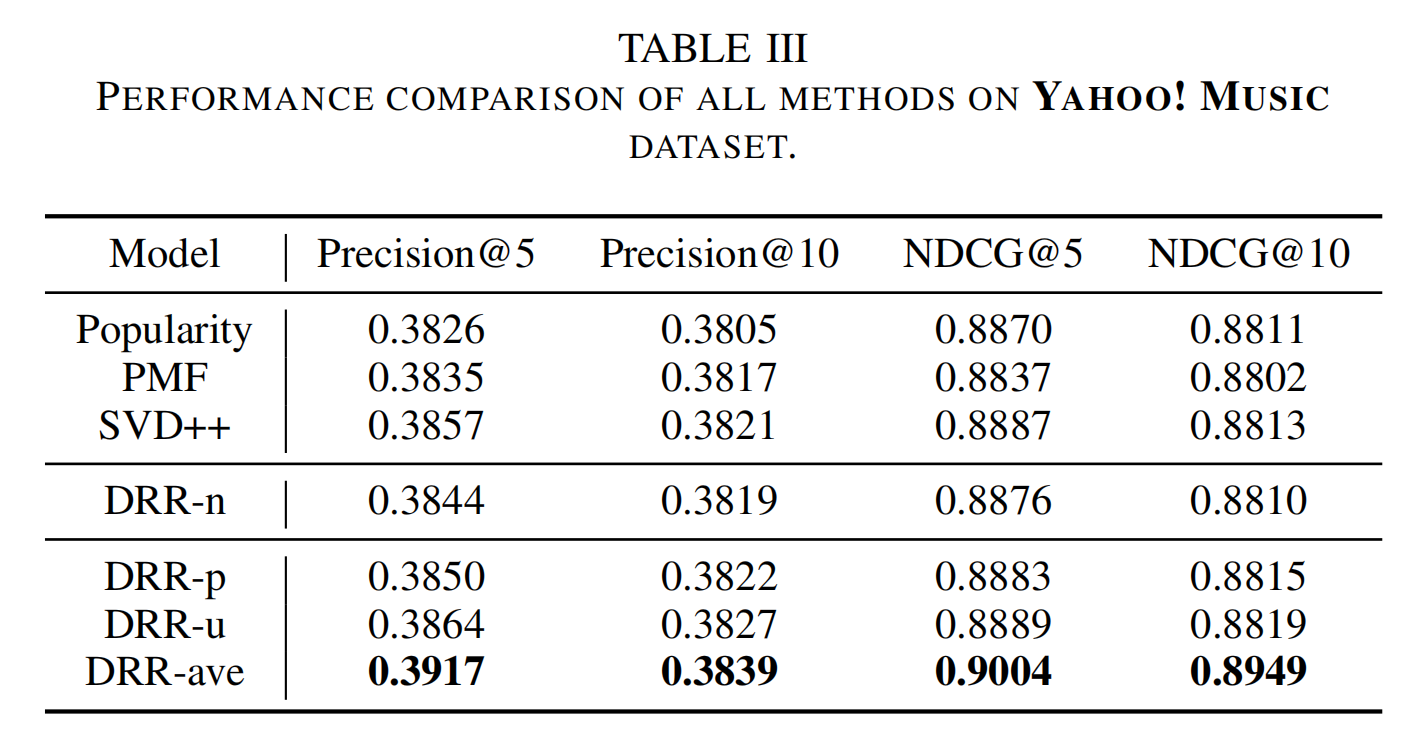
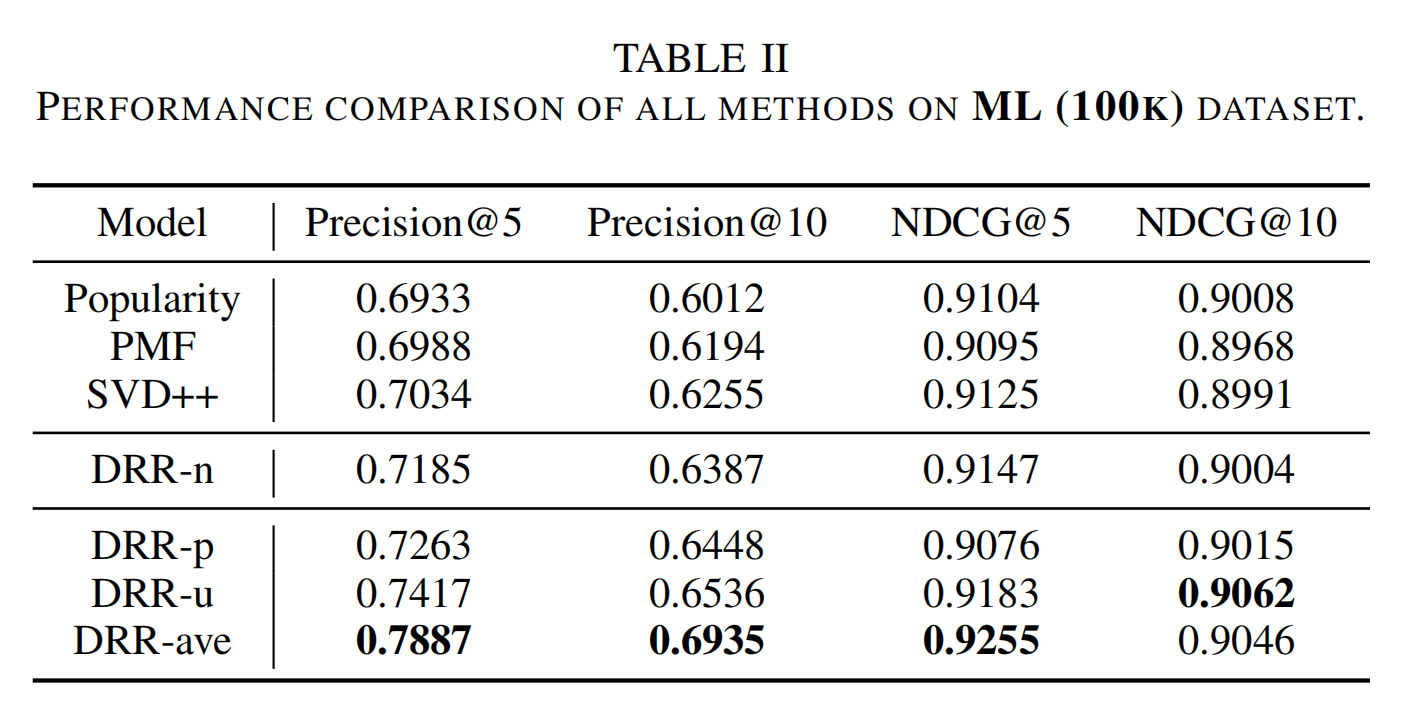
where the first setting is for MovieLens (100k), Yahoo! Music and MovieLens (1M), and the second one is for Jester. All the baseline methods are carefully tuned for a fair comparison. We model the recommendation procedure as an interaction episode with length T, and the hyper-parameter T is tuned for different datasets (detailed in Section V.E).

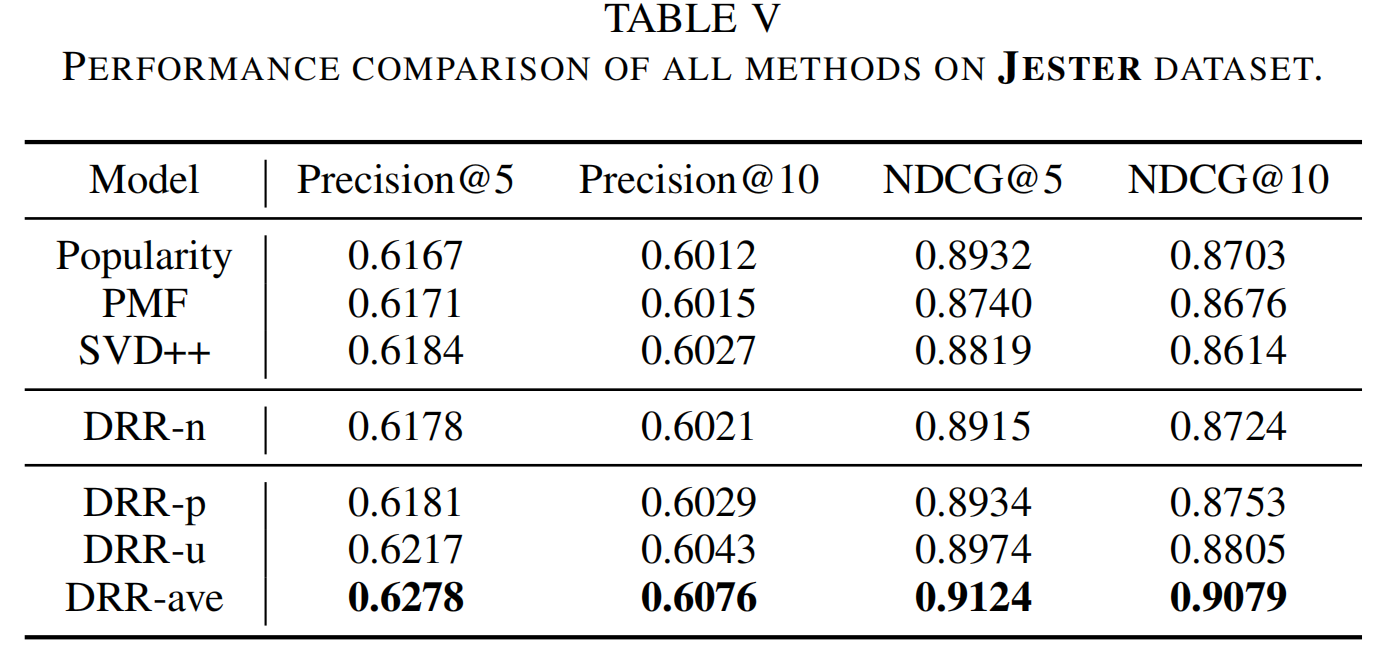
其中第一个设置是针对MovieLens（100k）的Yahoo! Music and MovieLens（1M），第二个是供Jester使用。 所有基线方法均经过仔细调整，以进行公平比较。 我们将推荐过程建模为长度为T的交互事件，并针对不同的数据集调整了超参数T（在第V.E节中进行了详细介绍）。

D. Results and Analysis

1) Offline Evaluation Results and Analysis: The offline evaluation results are summarized from Table II to Table V respectively, where the best results are marked in bold type. In the offline evaluation, we compare the proposed methods to some representative offline learning methods. The results suggest that the proposed methods under the DRR framework outperform the baselines on most of datasets, which demonstrates the effectiveness of our proposed methods.

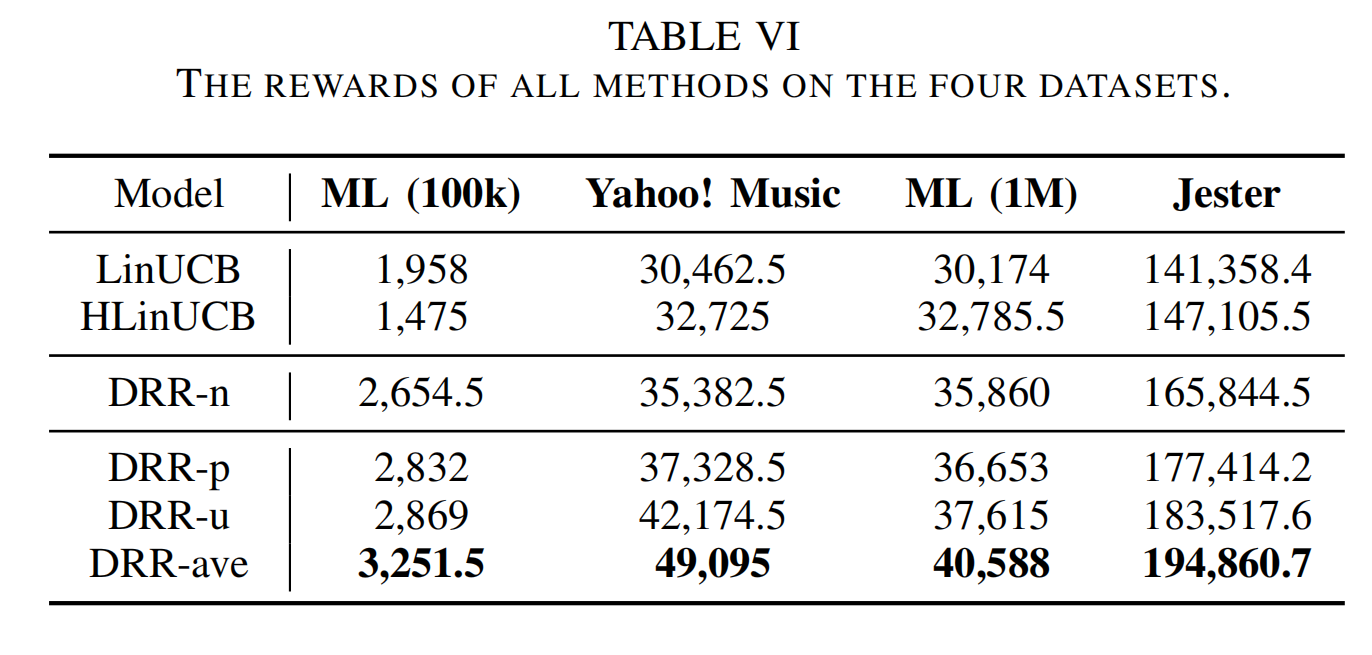
1）离线评估结果和分析：离线评估结果分别总结于表II至表V，其中最佳结果以粗体标出。 在离线评估中，我们将提出的方法与一些代表性的离线学习方法进行了比较。 结果表明，在DRR框架下提出的方法优于大多数数据集的基线，这证明了我们提出的方法的有效性。





2) Simulated online evaluation results and analysis: The results of the simulated online evaluation are summarized in Table VI, where the best results are marked in bold type. In the experiment, we only compare with the baseline methods that can perform online learning, which are LinUCB, HLinUCB and DRR-n. Again, we find that the proposed methods deliver higher rewards than all the baselines.

2）模拟在线评估结果和分析：模拟在线评估的结果总结在表VI中，其中最佳结果以粗体标出。 在实验中，我们仅与可以执行在线学习的基准方法（LinUCB，HLinUCB和DRR-n）进行比较。 同样，我们发现所提出的方法比所有基线提供更高的回报。



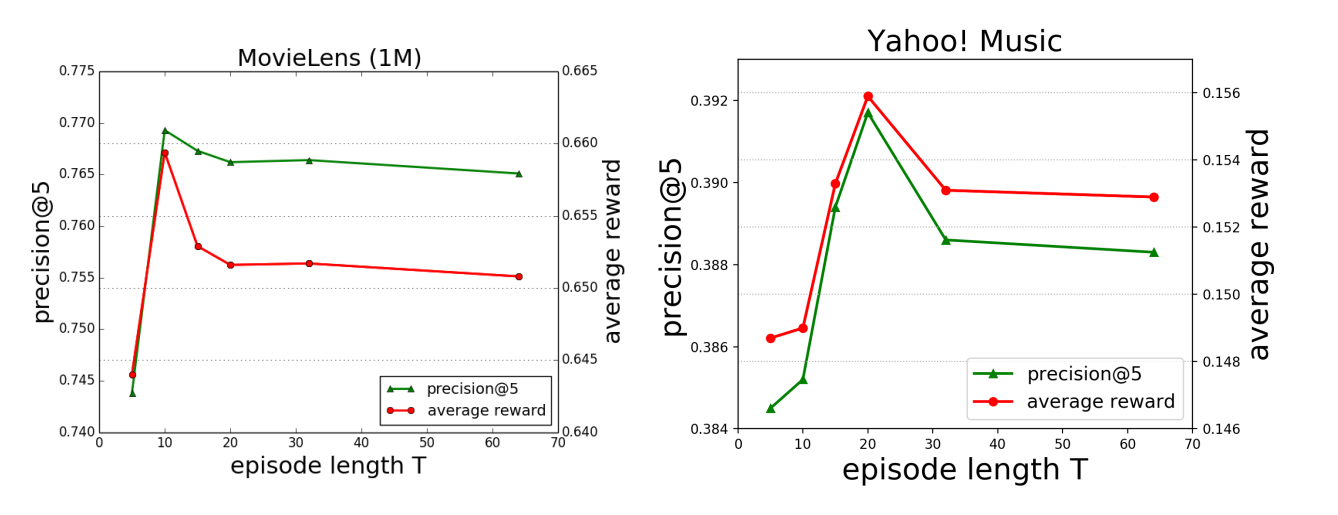
On the one hand, the fact suggests that the proposed RL-based methods model dynamic adaptation and long-term rewards better than the multi-armed bandits based methods LinUCB and HLinUCB. On the other hand, the observation indicates that the proposed state representation structures are superior to the naive full-connected network in DRR-n. Again, we observe that DRR-ave performs the best among all the three proposed interaction modeling structures.

一方面，事实表明，与基于多臂赌博机的方法LinUCB和HLinUCB相比，所建议的基于RL的方法对动态适应和长期奖励进行了建模。 另一方面，观察结果表明，所提出的状态表示结构优于DRR-n中的朴素全连接网络。 再次，我们观察到在所有三个建议的交互建模结构中，DRR-ave表现最佳。

1. *Parameter Study*

In this subsection, we investigate how the episode length T affect the performance of proposed methods. Figure 7 shows the results9 . From the left part of Figure 7, we observe that the performance on MovieLens first increases and then decreases as the length of the episode is gradually increased, and the summit appears at T = 10. A similar tend can be found for the Yahoo! Music from the right part of Figure 7, where the performance peaks at T = 20. The reason may due to the trade-off between the exploitation and exploration. When the episode length is small, the user can not fully interact with the recommender agent, i.e., the exploration is insufficient. As we enlarge the episodes, the recommender agent can explore (interact with users) adequately, i.e., the recommender agent captures the user’s preference, so that the performance improves. However, if the episodes are too large, the recommender focuses on exploiting locally, but the user preferred items is limited, therefore the performance declines as we do not recommend repeated items to user. Hence, we should nicely trade off the exploration and exploitation by setting a suitable value for T.

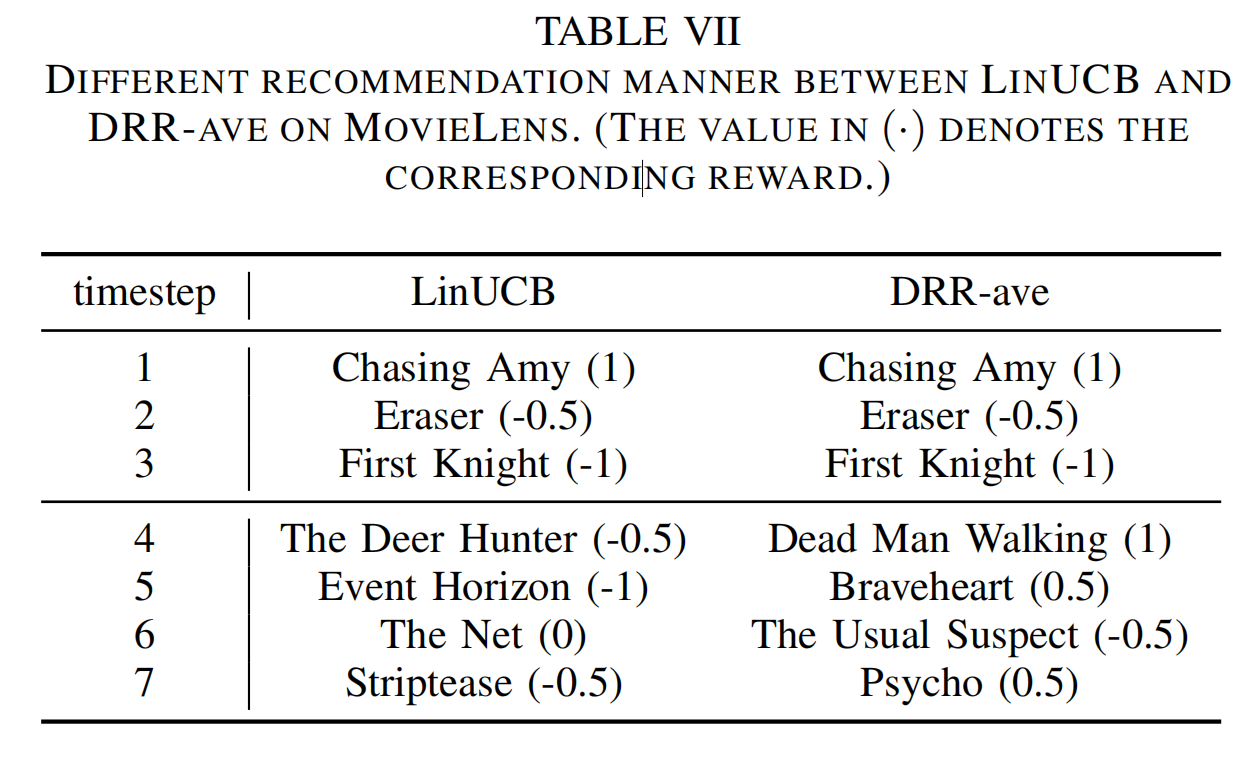
在本小节中，我们将研究幕长度T如何影响所提出方法的性能。图7显示了结果9。从图7的左侧开始，我们观察到MovieLens的性能随着幕的长度逐渐增加而先增加然后降低，并且峰顶出现在T =10。对于Yahoo!可以发现类似的趋势。图7右侧的音乐表现在T = 20时达到最高峰。其原因可能是开发与探索之间的权衡。当幕长度较小时，用户不能与推荐者代理完全交互，即，探索不足。当我们扩大剧集时，推荐人代理可以充分探索（与用户互动），即，推荐人代理可以捕获用户的偏好，从而提高性能。但是，如果幕太大，推荐者将重点放在本地开发上，但是用户偏爱的项目受到限制，因此由于我们不向用户推荐重复的项目，因此性能下降。因此，我们应该通过为T设置合适的值来权衡勘探和开发。



1. Case Study

In this subsection, we present an example to show the different recommendation manner between LinUCB and DRRave on MovieLens dataset. Specifically, we randomly pick up a user with ID 11, and conduct the recommendation procedure with LinUCB and DRR-ave respectively. To verify the reaction to the same recommendation scenario, we fix the first three recommended items and to see what will happen next. The results of recommended item and the reward are reported in Table VII.

在本小节中，我们将提供一个示例来展示MovieLens数据集上LinUCB和DRRave之间的不同推荐方式。 具体来说，我们随机挑选ID为11的用户，并分别使用LinUCB和DRR-ave进行推荐过程。 为了验证对同一推荐方案的反应，我们固定了前三个推荐项目，然后看看接下来会发生什么。 推荐项目和奖励的结果列于表。



From Table VII, we can see that LinUCB and DRR-ave react differently when given two consecutive negative recommendations (Eraser and First Knight). Specifically, LinUCB keeps exploring without considering to recommend a “safe” item to please the user. However, DRR-ave stops exploration and recommends a risk-free movie Dead Man Walking, which belongs to the same genre as Chasing Amy that has gained a positive feedback from the user at timestep 1. The observation demonstrates the superiority of the proposed DRR-ave against LinUCB.

从表VII中，我们可以看出，当连续两次给出负面建议（橡皮擦和第一骑士）时，LinUCB和DRR-ave的反应有所不同。 具体来说，LinUCB一直在探索而未考虑推荐“安全”物品来取悦用户。 但是，DRR-ave停止了探索并推荐了一部无风险的电影，与《追逐艾米》属于同一类型的电影，该电影在第1步获得了用户的积极反馈。观察结果表明，提出的DRR- 反对LinUCB。

VI. CONCLUSION

In this paper, we propose a deep reinforcement learning based framework DRR to perform the recommendation task. Unlike the conventional studies, DRR treats the recommendation as a sequential decision making process and adopts an “Actor-Critic” learning scheme, which can take both the immediate and long-term rewards into account. In DRR, a state representation module is incorporated and three instantiation structures are designed, which can explicitly model the interactions between users and items. Extensive experiments on four real-world datasets demonstrate the superiority of the proposed DRR method over state-of-the-art competitors.

在本文中，我们提出了一种基于深度强化学习的框架DRR来执行推荐任务。 与常规研究不同，DRR将推荐视为顺序决策过程，并采用“演员-批评”学习方案，该方案可以考虑近期和长期的奖励。 在DRR中，并入了一个状态表示模块，并设计了三个实例化结构，它们可以显式地对用户和项目之间的交互进行建模。 在四个真实世界的数据集上进行的广泛实验表明，提出的DRR方法优于最新的竞争对手。