information

MDPI

Article

Exploring Clustering-Based Reinforcement Learning for Personalized Book Recommendation in Digital Library

Xinhua Wang [1], Yuchen Wang [1,*], Lei Guo [2,*], Liancheng Xu [1], Baozhong Gao [1], Fangai Liu [1] and Wei Li [3]

[1] School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China; wangxinhua@sdnu.edu.cn (X.W.); lchxu@163.com (L.X.); gaobaozhong@sdnu.edu.cn (B.G.); lfa@sdnu.edu.cn (F.L.)
[2] School of Business, Shandong Normal University, Jinan 250358, China
[3] Library, Shandong Normal University, Jinan 250358, China; liww72@sdnu.edu.cn
* Correspondence: hermitwyc@gmail.com (Y.W.); guolei@sdnu.edu.cn (L.G.)

Abstract: Digital library as one of the most important ways in helping students acquire professional knowledge and improve their professional level has gained great attention in recent years. However, its large collection (especially the book resources) hinders students from finding the resources that they are interested in. To overcome this challenge, many researchers have already turned to recommendation algorithms. Compared with traditional recommendation tasks, in the digital library, there are two challenges in book recommendation problems. The first is that users may borrow books that they are not interested in (i.e., noisy borrowing behaviours), such as borrowing books for classmates. The second is that the number of books in a digital library is usually very large, which means one student can only borrow a small set of books in history (i.e., data sparsity issue). As the noisy interactions in students' borrowing sequences may harm the recommendation performance of a book recommender, we focus on refining recommendations via filtering out data noises. Moreover, due to the the lack of direct supervision information, we treat noise filtering in sequences as a decision-making process and innovatively introduce a reinforcement learning method as our recommendation framework. Furthermore, to overcome the sparsity issue of students' borrowing behaviours, a clustering-based reinforcement learning algorithm is further developed. Experimental results on two real-world datasets demonstrate the superiority of our proposed method compared with several state-of-the-art recommendation methods.

Keywords: reinforcement learning; recommender system; clustering; book recommendation

Citation: Wang, X.; Wang, Y.; Guo, L.; Xu, L.; Gao, B.; Liu, F.; Li, W. Exploring Clustering-Based Reinforcement Learning for Personalized Book Recommendation in Digital Library. *Information* **2021**, *12*, 198. https://doi.org/10.3390/info12050198

Academic Editor: José J. Pazos Arias

Received: 5 April 2021
Accepted: 26 April 2021
Published: 30 April 2021

1. Introduction

Digital library as one of the most important ways in helping students acquire professional knowledge and improve their professional level has gained great attention recently. Many universities have established their digital libraries with digital resource ranges from tens of thousands to millions. On the one hand, the digital library is more convenient in use and management than traditional libraries. On the other hand, it is a huge challenge for students to find the required resources (such as books, reports, and periodicals). To overcome this challenge, we resort to recommender systems [1–4], which can leverage users' historical records to help them efficiently discover interesting and high-quality information. The book recommendation task in a digital library is to recommend books at time $t+1$ to a set of users given the historical book borrowing records before time t.

Typical works of previous research on this task are focused on developing recommendation algorithms that can recommend books in a personalized way. For example, Yang et al. [5] introduced a book recommender system for book inquiry history analysis and proposed a model for book inquiry history analysis and book-acquisition recommendation. Sohail et al. [6] proposed an OWA-Based ranking approach for the book

recommendation. They used an ordered weighted aggregation method to aggregate several rankings of top universities, which reduces the complexity of providing personalized recommendations to a large number of users. Priyanka et al. [7] developed a personalized book recommender system based on opinion mining technique and presented an online book recommender system for users who purchase books by considering the specific functions of books.

However, all the above existing methods ignored a widely reached consensus that the noisy data can mislead the recommendation algorithms. For example, a student studying computer science and technology may have borrowed few psychological books from the library for public elective courses or other students. When recommending books to him/her, we should mainly focus on recommending books related to computer science, and ignore the influence of these psychological books. We define the items in the user's borrowing record that the user is not interested in as noise in the user's borrowing record. Moreover, all the above methods ignored another fact that the users' borrowing records are very sparse, that is, a user may only borrow a few books in his/her college years, which results in that the learned user interest model is unreliable. We define the small amount of user borrowing and the small amount of each book borrowed as the data sparseness problem in the book recommendation task. Noise problem and user sparsity problem are two problems that need to be solved in the book recommendation.

On the other hand, reinforcement learning-based recommender system has achieved great success in many other related areas, such as course recommendation in MOOC platforms and TV program recommendations. For example, Zhang et al. [8] developed a Hierarchical Reinforcement Learning(HRL) framework to alleviate the impact of noisy user behaviors in course recommendation task(HRL-NAIS), where a hierarchical agent is developed to filter the noisy actions that might mislead the recommendation algorithm. However, their work is developed for the course recommendation in MOOC platforms rather than book recommendation in digital libraries. Compared with course recommendation, book recommendation is significantly different from it. First, there are a huge number of books in the library. The types of books in the book recommendation are much more than the types of courses in the course recommendation. Second, the average amount of books borrowed by users in the digital library is small. Compared with the average subscription amount in the course recommendation, the average borrowing amount of the users in the digital library is only half of the course recommendation, which means the data in book recommendation is sparser, and each book will only be borrowed a few times on average, while in course recommendation each course may be subscribed hundreds of times on average.

To overcome the above challenges, we introduce HRL into book recommendation task, and further propose a clustering-based HRL in dealing with the sparsity issue of users' interactions. More specifically, to filter out the noisy data that existed in users' borrowing sequence, we train a hierarchical reinforcement network to select the books that can help the recommender most. To alleviate the sparsity issue of users' interactions, we cluster the feature of the books before feeding the book data into the hierarchical reinforcement network, so that the hierarchical reinforcement network can better.

The main contributions of this work are listed as follows:

1. We introduce HRL into the book recommendation task in the digital library, where a basic book recommender is first pre-trained, and then a hierarchical agent is devised to filter out the interactions that might miss leading this recommender.
2. To reduce the impact of the sparsity issue, we further enhanced the HRL by a clustering-based strategy, where a clustering strategy between the pre-trained network and the hierarchical reinforcement network is incorporated to reduce the data sparsity issue of the book borrowing data.
3. We conduct extensive experiments on two real-world datasets, and the experimental results demonstrate the superiority of our solution compared with several state-of-the-art recommendation methods.

2. Related Works

This section reviews the related works from the following aspects, that is, book recommendation methods and reinforcement learning-based recommendation methods.

2.1. Book Recommendation Methods

Book recommendation as an important branch of recommender system has been widely studied in recent years. Its target is to recommend suitable books to users that have interests in them. Existing recommendation methods can be classified into collaborative filtering-based methods and general machine learning-based methods. For example, Ansari et al. [9] studied the advantages of content-based and collaborative filtering methods, proposed a preference model used in marketing to provide a good choice, and described a Bayesian preference model. Ziegler et al. [10] proposed diversification of topics, which is a new method that aims to balance and diversify personalized recommendation lists to reflect the complete range of interests of users. Konstas et al. [11] created a collaborative recommendation system that can effectively adapt to the personal information needs of each user. They adopted a general framework of random walks with restarts to provide a more natural and effective way to represent social networks. Robillard et al. [12] proposed a software engineering recommendation system that can help developers in various activities, from reusing code to writing effective error reports. Smyth et al. [13] proposed a case-based recommendation system based on a well-defined feature set (such as price, color, brand, etc.). These representations enable case-based recommenders to make judgments on the similarity of products to improve the quality of their recommendations. Fu et al. [14] captures the user's navigation history and applies data mining techniques to discover the hidden knowledge contained in the history. This knowledge will then be used to suggest potentially interesting web pages to the user. Drineas et al. [15] put forward the concept of a competitive recommendation system. They simplified the problem of obtaining competitiveness into a problem in matrix reconstruction. Then a competitive matrix reconstruction scheme is proposed. The above methods solve the recommendation problem based on collaborative filtering. However, methods based on collaborative filtering have certain limitations: They tend to recommend popular items in collaborative filtering algorithms, so the algorithm is less exploratory. In addition, the algorithm cannot perform well when the data is sparse. So with the popularity of machine learning, more and more machine learning methods are used in recommendation systems.

Besides traditional collaborative filtering methods, many researches also focused on generalized machine learning-based methods. For example, Sabitha et al. [16] proposed a book recommendation method based on user NN. In order to optimize collaborative filtering, Goel et al. [17] proposed a bee algorithm that uses natural inspiration. Natural language processing in machine learning is applied in the algorithm. Mikawa et al. [18] used the Support Vector Machine(SVM) algorithm in the book recommendation. SVM is a kernel-based technique that is used for the categorization of binary data. The model in the article recognizes the gender of the mobile person and recommends magazines to the person based on gender. The model shows a high recommended speed. The neural network model is inspired by the human brain, constituted by the connection between the node and another neuron. This method is used in the model built by liu et al. [19], which proves that the blending neural network outperforms the blending with Linear Regression. Maneewongvatana et al. [20] used k-means clustering to recommend books in university libraries. They browsed the library's borrowing history, and after cleaning the data, they assigned them to different clusters based on the similarity of the topics. Then implemented association rule mining to recommend books. Yang et al. [21] used text mining technology to establish a recommendation model for book purchases. They use three modules: keyword density vocabulary, keyword sequence vocabulary, and keyword book mapping, and establishes extraction of keywords, matching with keywords in the book library to obtain recommendation books, Recommendation system that generates recommendation lists. Tewari et al. [22] proposed a method of recommending books to readers. The system

combines the functions of content filtering, collaborative filtering, and association rule mining to generate effective suggestions. Sohail et al. [23] proposed a recommendation technique based on opinion mining. According to customer needs and comments collected from customers, the article classifies the functions of the books. The article analyzes the functions based on several features that have been classified and commented on by users. According to the importance and purpose of the weight, the weight is assigned to the classified features, and the grade is given accordingly. Kanetkar et al. [24] proposed a model of a Web-based personalized hybrid book recommender system, which uses various aspects other than conventional collaboration and content-based filtering methods to provide recommendations. Vaz et al. [25] proposed a hybrid recommendation task that combines two item-based collaborative filtering algorithms to predict users' favorite books and authors. The author's forecast was expanded into a bibliographic list, which was then aggregated with previous book forecasts. Finally, the generated book list will be used to generate the top n book recommendations. The use of machine learning methods can use the characteristics of books to classify books, which can solve the problem of too many popular items recommended in the collaborative filtering method. The clustering method in deep learning can also solve the problem of data sparsity in the recommendation system.

However, the above methods did not consider the noisy data that existed in users' book borrowing records, which may mislead the recommendation methods. We need a new method that can remove the noise in the sequence.

2.2. Deep Reinforcement Learning

Deep reinforcement learning is a machine learning method that combines reinforcement learning and deep learning. The characteristic of reinforcement learning is to map the current state to certain actions, and then select the best action based on the return value of these actions. The characteristic of deep learning is to obtain observation information from the environment. Combining the characteristics of reinforcement learning and deep learning can play the decision-making ability of reinforcement learning and the perceived ability of deep learning, and provide ideas for complex perception decision-making problems.

Mnih et al. [26] first proposed the DQN algorithm, for the first time combining deep learning with reinforcement learning. The DQN algorithm puts the calculation of the Q value in reinforcement learning into the deep learning model for calculation. In the following years, Hausknecht and Stone [27] proposed the Deep Recurrent Q-Learning algorithm, Wang et al. [28] proposed the Dueling DQN algorithm, Van et al. [29] proposed the Double DQN algorithm, Schaul et al. [30] proposed the Prioritized Experience Replay algorithm, Hessel et al. [31] proposed Rainbow DQN algorithm, these algorithms are based on the improvements made by DQN algorithm.

In the reinforcement learning algorithm based on DQN, the algorithm is based on value to learn. Among the deep reinforcement learning algorithms, there is another type of algorithm that is based on policy to learn. The A3C algorithm proposed by Mnih et al. [32] is based on the policy gradient algorithm. The A3C contains an actor-network and an evaluator network and selects the optimal action for training through the two networks.

The hierarchical reinforcement learning model is a reinforcement learning algorithm that can handle complex environments. The hierarchical reinforcement learning model can divide the entire task into high-level tasks and low-level tasks. The high-level tasks plan long-term benefits, and the low-level tasks directly interact with the environment. There are several articles on the research of the hierarchical reinforcement learning model: Vezhnevets et al. [33] proposed the STRAW algorithm, and then proposed the Feudal Networks algorithm [34], Nachum et al. [35] proposed the HIRO algorithm. The hierarchical reinforcement learning model can effectively deal with sparse rewards, so it can be effectively used in the recommendation system.

2.3. Reinforcement Learning-Based Recommendation Methods

The traditional recommendation task analyzes the user's actions and recommends similar content to the user. This recommendation has the following problems: It is unable to capture the dynamic changes of user interest; it can only calculate the user's current revenue; it is difficult to accumulate long-term revenue. One way to solve those problems is to introduce reinforcement learning into the recommendation task.

Reinforcement learning is the mapping relationship between learning state and behavior to maximize the numerical return. In other words, without knowing what behavior to take, the learner must keep trying to find out which behavior can produce the greatest reward. Theocharous et al. [36] proposed the use of reinforcement learning in advertising recommendations, pointed out the benefits of reinforcement learning in recommendation systems: Considering the long-term effects of actions. The paper also illustrates the challenges of reinforcement learning in recommendation systems: how to calculate good strategies and how to use historical data to evaluate solutions. Wang et al. [37] proposed to apply reinforcement learning to music recommendation. This paper balances the needs of exploring user preferences and of using this information to make recommendations. To learn user preferences, this paper uses a Bayesian model, which considers the novelty of audio content and recommendations, and regards music recommendations and playlist generation as a unified Model. Zheng et al. [38] proposed the application of reinforcement learning to news recommendation, and proposed a recommendation framework based on deep Q-learning, which solves three problems of news recommendation: It only attempts to simulate the current rewards; it rarely studies except click and no click Operation; it tends to recommend similar news. Wang et al. [39] proposed a dynamic therapy recommendation task based on large-scale electronic health records. The paper combines the benefits of supervised learning and reinforcement learning and proposes supervised reinforcement learning with regression neural networks. The paper uses an actor-critic framework to deal with the complex relationship between multiple drugs, diseases, and individual characteristics, which can provide effective advice to doctors. Zhao et al. [40] focused on dealing with the negative feedback of reinforcement learning in the recommendation task, developed a deep recommendation task framework, and combined negative feedback with positive feedback. Zhao et al. [41] combined reinforcement learning, recurrent neural network, and convolutional neural network to design the model, and used the model in e-commerce recommendation to improve the accuracy of e-commerce recommendation. Chen et al. [42] proposed two techniques to alleviate the problem of unstable reward estimation in dynamic environments. The article addresses the problem of high variance and deviation estimation of rewards in terms of samples and rewards respectively and integrates these two techniques with deep reinforcement learning. Rohde et al. [43] introduced RecoGym, a reinforcement learning environment defined by the user traffic model in e-commerce and the user's recommendation on the website. It can be seen from the above research that the application of reinforcement learning in the recommendation task can consider the long-term impact of each action of the user, and it can also solve the problem of single recommendation content, and solves the problem that the traditional recommendation task cannot model the dynamic changes of user interest.

The recommendation system model based on reinforcement learning can better capture the dynamic changes of user interests, and the noise in the sequence can be processed by introducing reinforcement learning. However, because book data is relatively sparse, and these works did not consider the problem of data sparseness, we need to establish a recommendation system model based on reinforcement learning suitable for book data.

3. Our Recommendation Framework

The section first defines the book recommendation task and then introduces the HRL framework to filter the noisy data exited in users' historical interactions. Finally, our clustering-based reinforcement learning method is introduced in detail.

3.1. Task Definition

The task of book recommendation for a user can be simply formalized as next item prediction based on his/her historical borrowing records, where the borrowing sequence before time t are given, and we aim at recommending the most relevant books that will be enrolled by the user at time $t+1$.

Let $\mathcal{U} = \{u_1, \ldots, u_m\}$ be the set of users and $\mathcal{B} = \{b_1, \ldots, b_n\}$ be the set of books, where m is the number of users, n is the number of books. For each user $u \in \mathcal{U}$, his/her borrowing sequence $E^u := (b_1^u, \ldots, b_j^u, \ldots, b_t^u)$ from a digital library is given, where t denotes the time stamp of the corresponding record. Then, the book recommendation task can be formulated as predicting the next book that the user is interested.

Input: The set of users $\mathcal{U} = \{u_1, \ldots, u_m\}$, the set of books $\mathcal{B} = \{b_1, \ldots, b_n\}$, and each user's borrowing sequence $E^u := (b_1^u, \ldots, b_j^u, \ldots, b_t^u)$.

Output: The next book b_{t+1}^u that the user u is most interested in.

3.2. Overview of Our CHRL Method

To solve the data noisy and data sparseness challenges in book recommendation, we propose a Clustering-based Hierarchical Reinforcement Learning Network (CHRL) as our solution, whose main idea is to leverage the power of the clustering-based reinforcement learning technique to filter out noisy interactions that may mislead the recommendation algorithms. Figure 1 shows the architecture of our model, which consists of three components: Basic recommender, profile reviser, and clustering component. More specifically, the basic recommender aims to provide the foundation model of book recommendation, which models user's and item's preferences by an attention-based neural network. Same as HRL [8], we exploit NAIS [44] as our basic model. Note that, other sequence modeling techniques can also be utilized. The profile reviser aims to further filter out noisy interactions that may mislead the basic recommender. A clustering-based hierarchical reinforcement learning network is developed. Different the work in HRL [8], we develop a clustering component to alleviate the data sparsity issue via clustering the embedding of all the books to categorize the books.

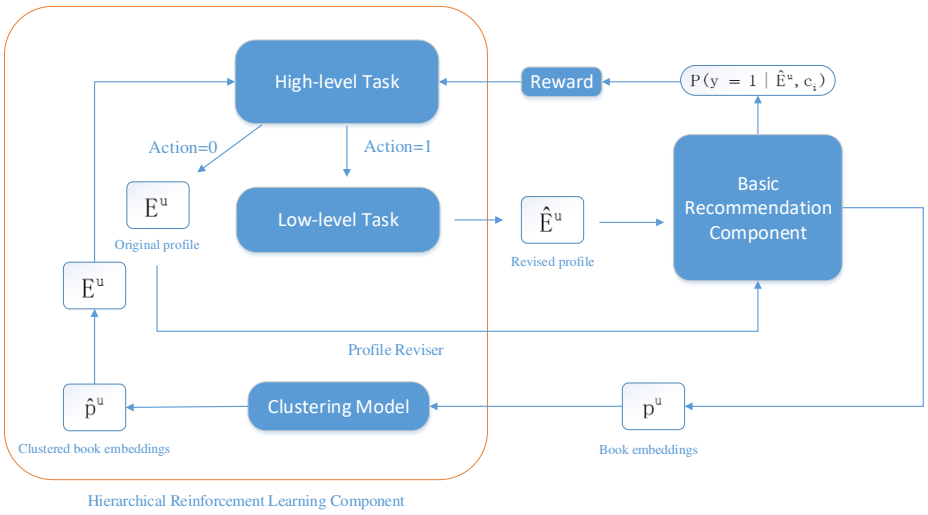

Figure 1. Overview of our CHRL Model.

The workflow of our CHRL model is shown in Figure 2. First, the pre-training process of CHRL trains the borrowing sequence of all users. After pre-training, the clustering component clusters the embedding of all the learned books that will be utilized in the sequence modification component. Next, the sequence modification component will determine whether there are noises in the borrowing sequence of each user and then filter out these noisy interactions. Finally, the selected sequences will be re-sent to the pre-trained

model in the first stage. After that, the pre-trained basic recommender and HRL will be jointly trained to obtain the optimal recommendation results. Our CHRL uses HRL to filter out the noisy data in the sequence, and resort to the clustering mechanism to alleviate the data sparsity issue of book recommendation.

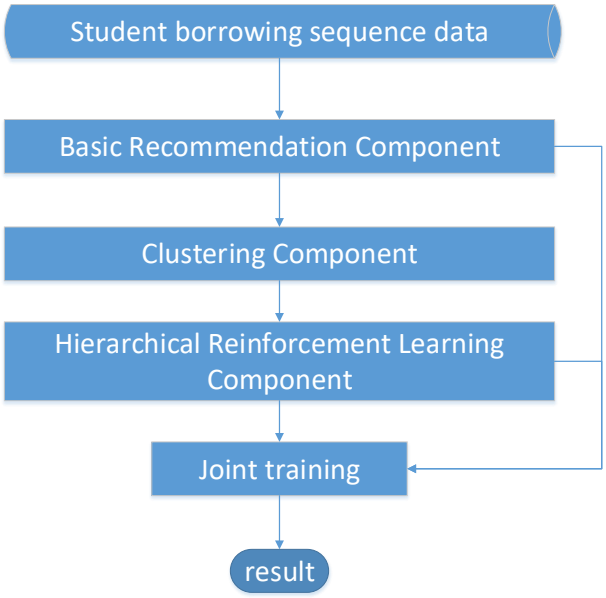

Figure 2. The workflow of our CHRL model.

3.3. Basic Recommender

To model user's borrowing sequences, we follow the work in HRL [8] and exploit NAIS [44] as our basic sequence modeling method. The NAIS algorithm can effectively solve the learning problem based on item-to-item collaborative filtering. It represents an item as an embedding vector and models the similarity between two items as the inner product of their embedding vectors, so a sequence can be expressed as a set of embedding vectors, and the results can be predicted using the embeddings of each item. On this basis, NAIS used the attention network to learn the different importance of interactive items.

As NAIS does, to characterize a user's preference according to his borrowing sequence E^u, we represent each historical book as a real-valued low dimensional embedding vector $\boldsymbol{p}_t^u$. Then, we can express the student's borrowing sequence as $\boldsymbol{p}_1^u, \ldots, \boldsymbol{p}_t^u$, and denote a target book b_i via an embedding vector $\boldsymbol{p}_i$. Moreover, we represent the student borrowing sequence by $\boldsymbol{q}_u$, and aim to calculate the probability of recommending book b_i to students. When representing the features of the sequence $\boldsymbol{q}_u$, we use the attention mechanism and add an attention factor to each element $\boldsymbol{p}_t^u$ in the sequence $\boldsymbol{q}_u$, which can more clearly indicate students' different interests in each book.

3.4. The Sequence Modification Component

To filter out the noisy interactions from users' borrowing sequences, we resort to the reinforcement learning technique, which can transform the process of modifying the borrowing sequence into a hierarchical Markov decision process. Motivated by HRL [8], we also treat the process of modifying the student borrowing sequence as a hierarchical Markov decision process M, and divide it into two steps, high-level task M_h and low-level task M_l. The high-level task M_h determines whether the entire sequence needs to be modified. If it needs to be modified, it enters the low-level task M_l. The low-level task M_l determines whether each element in the sequence should be deleted. After modifying the sequence, the agent will give a delayed reward according to the environment and the modified sequence.

Clustering component. While the method HRL [8] has obtained excellent results in the sequential recommendation, their work is not suitable for the book recommendation due to its data sparsity issue. In a digital library, the borrowing histories of students are very spare, that is, most of the users only borrow a few books. In this condition, the feature distribution of the data is relatively scattered. The high-level tasks and low-level tasks of hierarchical reinforcement learning tasks tend to delete all interactions, causing the basic recommender to fail to make correct recommendations. Therefore, after the pre-training of the basic recommender, we devise a clustering component to cluster the embedding of all the books used in the HRL component to make it more stable.

After the basic model is trained, each book gets its embedding. As our goal is to allow the sequence modification component to more accurately identify the noisy books, rather than deleting all the elements in the borrowing sequence, we need to cluster the embedding of the books. After clustering, the embedding $\boldsymbol{p}$ of each book becomes the embedding $\boldsymbol{p}_c$ of the center point corresponding to this embedding. In this paper, the embedding of the book with the smallest serial number in each category is taken as the center point of the category. In other words, after clustering, the embedding of each book will become the center point embedding $\boldsymbol{p}_c$ of this category, such as shown in Figure 3. We represent the borrowed book sequence E^u after clustering as E^u_c. The sequence modification component will use the clustered embedding for training and modify the elements in the borrowing sequence.

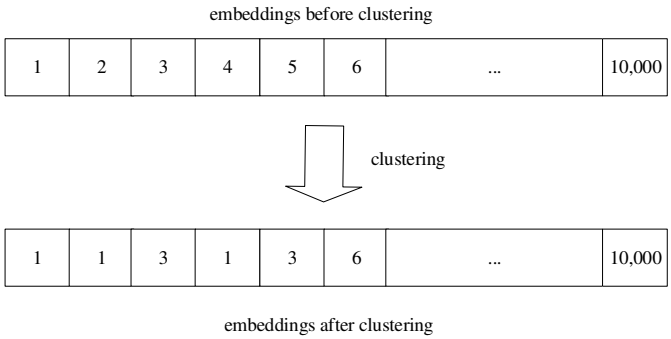

Figure 3. The process of clustering embeddings.

In the following, we will describe the key definitions used in our reinforcement learning component: Environment, state, action, decision-making, and reward [8].

Environment: The environment is regarded as the dataset of books and the pre-trained basic recommender.

State: In the high-level task, the high-level task determines whether the entire borrowing sequence needs to be modified, and the low-level task determines whether each borrowing record in the sequence needs to be deleted. The state of the low-level task is defined as the cosine similarity between the current borrowing sequence and the target book's embedding vector. The state of the high-level task is defined as the average cosine similarity and the average element-wise product between the embedding vector of each borrowed record in the borrowing sequence and the embedding vector of the target book. Besides, the basic recommendation model recommends the target book based on the probability value of the borrowing sequence to reflect the credibility of the target book. If the credibility is lower, the borrowing sequence should be modified.

Action and Policy: In high-level task, we define action a^h as $a^h \in \{0, 1\}$, action a^h is a binary value that indicates whether to enter the low-level task and modify the borrowing sequence. The action $a^l_t \in \{0, 1\}$ in the low-level task is also a binary value, indicating whether to delete this borrowing record. We perform a low-level action a^l_t according to the policy function as follows:

$$H^l_t = ReLU(W^l_1 s^l_t + b^l) \tag{1}$$

$$\pi(s^l_t, a^l_t) = P(a^l_t | s^l_t, \Theta^l) = a^l_t \sigma(W^l_2 H^l_t) + (1 - a^l_t)(1 - \sigma(W^l_2 H^l_t)) \tag{2}$$

where $W_1^l \in R^{(d_1^l \times d_2^l)}$, $W_2^l \in R^{(d_1^2 \times 1)}$, $b^l \in R^{(d_2^l)}$ are the parameters to be learned, d_1^l is the number of the state features and d_2^l is the dimension of the hidden layer. H_t^l is the embedding of the input state. We denote the parameter to be learned as $\Theta^l = \{W_1^l, W_2^l, b^l\}$. σ is a sigmoid function, and it converts the input into probability. For high-level tasks, the policy function is similar to the low-level tasks, just change the parameter to $\Theta^h = \{W_1^h, W_2^l, b^h\}$.

Reward: The reward represents whether the action performed is reasonable. For low-level tasks, assuming that each action in the low-level task process has a delayed reward for the last action in the process, then the reward of each low-level action is defined as:

$$R(a_t^l, s_t^l) = \begin{cases} \log p(\hat{E}_c^u, c_i) - \log p(E_c^u, c_i) & \text{if} \quad t = t_u \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $p(E^u, c_i)$ is an abbreviation of $p(y = 1|E_c^u, c_i)$ and $\hat{E}_c^u$ is the modified sequence after clustering. In the process of performing low-level tasks, the agent may delete all elements in the sequence. At this time, the model randomly selects an element from the sequence as the modified sequence. When you perform a high-level task if the high-level task chooses to modify this sequence, then the reward of the high-level task is the same as the reward of the corresponding low-level task, and if you choose not to modify it, the reward is zero.

In addition, the model defines an internal reward G within the low-level tasks, and the purpose is to make the agent tend to choose the course most relevant to the target course. G is calculated as follows: The model calculates the average cosine similarity between each sequence element and the target book after and before the sequence is revised, and then uses their difference as the internal reward G.

Objective Function: As HRL does, we aim at finding the optimal parameters of the policy function to maximize the expected reward:

$$\Theta^* = argmax_\Theta \sum_\tau P_\Theta(\tau; \Theta) R(\tau) \tag{4}$$

where Θ represents either Θ^h or Θ^l, τ is a sequence of the sampled actions and the transited states, $P_\Theta(\tau; \Theta)$ denotes the corresponding sampling probability and $R(\tau)$ is the reward for the sampled sequence τ . The sampled sequence τ can be $\{s_1^l, a_1^l, s_2^l, \ldots, s_t^l, a_t^l, s_{t+1}^l, \ldots\}$ for the low-level task and $\{s^h, a^h\}$ for the high-level task.Then based on the trajectory K, we can calculate the gradient of the parameters of the low-level policy function:

$$\nabla_\theta = \frac{1}{k} \sum_{k=1}^{K} \sum_{t=1}^{t_u} \nabla_\theta \log \pi_\theta(s_t^k, a_t^k)(R(a_t^k, s_t^k) + G(a_t^k, s_t^k) \tag{5}$$

where the reward $R(a_t^k, s_t^k) + G(a_t^k, s_t^k)$ for each action state pair in sequence τ^k is assigned the same value and equals to the terminal reward $R(a_{t_u}^k, s_{t_u}^k) + G(a_{t_u}^k, s_{t_u}^k)$. we can also calculate the gradient for the high-level policy function:

$$\nabla_\theta = \frac{1}{k} \sum_{k=1}^{K} \nabla_\theta \log \pi_\theta(s^k, a^k) R(a_t^k, s_t^k) \tag{6}$$

where the reward $R(a^k, s^k)$ is assigned as $R(a_{t_u}^k, s_{t_u}^k)$ when a^k = 1 and 0.

3.5. Joint Training

Through hierarchical reinforcement learning, we got the revised sequences of students' borrowing history. As our task is to leverage these modified interactions to fun-tune the basic recommendation model to get more accurate recommendation results, we send these revised sequences back to the basic recommender, and re-train the sequence modification component and the basic recommender jointly. Algorithm 1 shows the joint training strategy of our proposed method.

Algorithm 1 The joint training strategy of CHRL

Input: Training sequence $\{E^1, E^2, \ldots, E^U\}$, $E^u = (b_1^u, \ldots, b_j^u, \ldots, b_t^u)$, parameters in the basic model Φ^0, parameters in the sequence modification model Θ^0.
Output: Prediction results for each sequence.
1. Initialization $\Phi = \Phi^0$, $\Theta = \Theta^0$.
2. Pretrain the basic model, get the embedding $\{p_1, p_2, \ldots, p_n\}$ of the book $\{b_1, b_2, \ldots, b_n\}$, update the network parameters Φ.
3. Cluster the embedding $\{p_1, p_2, \ldots, p_n\}$, get the clustered embedding $\{\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_n\}$, map the books $\{b_1, b_2, \ldots, b_n\}$ to the clustered embedding $\{\hat{b}_1, \hat{b}_2, \ldots, \hat{b}_n\}$, and get the clustered sequence $\{E_c^1, E_c^2, \ldots, E_c^U\}$.
4. For episode $l = 1$ to L do
    For each sequence $E^u = (\hat{b}_1^u, \hat{b}_2^u, \ldots, \hat{b}_t^u)$ and the book b_i
        Sample a high-level action a^h with Θ^h
        if $a^h = 0$ then
            $R(s^h, a^h) = 0$
        else
            Sample a sequence of low-level actions $a_1^l, a_2^l, \ldots, a_t^l$ with Θ^l
            Compute $R(a_t^l, s_t^l)$ and $G(a_t^l, s_t^l)$
            Compute gradients
    Update Θ
5. Jointly training, repeat the process of 2, 3, 4 steps to get the final output result.

4. Experiments

In this section, we first introduce the datasets we use, then describe the experimental environment and hyperparameters in our model, and finally give a brief introduction to our baseline.

4.1. Datasets and Experimental Settings

In experiments, we use two real-world datasets: School borrowing data from one digital library and Goodbooks (http://fastml.com/goodbooks-10k-a-new-dataset-for-book-recommendations/ accessed on 27 April 2021) to evaluate our proposed method.

School borrowing data contains 533,010 borrowing records. To ensure the recommendation quality, we filter out the students with less than 4 borrowing records and finally got 518,605 records, which are composed of 22,572 student borrowing sequences, containing 124,468 books. The statistics of this dataset are shown in Table 1.

As we can see from the data, the average number of books borrowed by each student is about 20, and most students borrow less than 20 books, but the number of books borrowed per book is only about 4 on average, which means that the book data is very sparse. If the common recommendation model is used to deal with sparse data, it is difficult for the model to accurately find books suitable for students. Therefore, our model has to do special processing on sparse data, so we use the clustering method to process the library data.

Another real-world dataset exploited in this work is the Goodbooks-10k dataset, which is a data set of book recommendations containing 912,705 reading records. After the same data processing, we finally get 898,195 reading records containing 10,000 books, and 41,314 users were finally obtained. The statistics of this dataset are shown in Table 1.

Data pre-processing. Since some users have only borrowed a small number of books, and a small number of data cannot form a sequence, we use the data of users whose number of borrowed books is greater than or equal to 4 in the two data sets. In the process of training and testing, we use the last item of these sequences as test data, and other items as training data to get the final result.

Table 1. Statistics of the Datasets.

	School Borrowing Data	Goodbooks-10k Data
Records	518,605	898,195
Students	22,572	41,314
Books	124,468	10,000
Training record	496,033	856,881
Testing record	22,572	41,314

4.2. Implementation Details

We implement CHRL using Python and TensorFlow library with NVidia RTX 2080 GPU. The basic model uses the NAIS model. We set the recommender epochs to 20, the learning rate to 0.02, and the embedding size in the model to 16. The number of categories of the clustering model is set to 2000. The agent epoch of the agent in the hierarchical reinforcement learning pre-training is set to 50, and the learning rate of hierarchical reinforcement learning pre-training is set to 0.05. At last, we set up joint training in learning rate and delayed coefficient of 0.05 and 0.0005. The specific parameter settings in the school borrowing data are shown in Table 2.

Goodbooks data and school borrowing data show similar characteristics. We adjust the number of categories to 1000 because the number of books in the Goodbooks-10k data is less than the number of books in the school data. The other hyperparameters are the same as the experiment on school data. The specific parameter settings in Goodbooks are shown in Table 2.

Table 2. Hyperparameters.

	School Borrowing Data	Goodbooks-10k Data
Basic model epochs	20	20
Basic model learning rate	0.02	0.02
Basic model embedding size	16	16
cateorties	2000	1000
pre-training epochs	50	50
pre-training learning rate	0.05	0.05
joint trainging learning rate	0.05	0.05
joint delayed coefficient	0.0005	0.0005

4.3. Baseline Methods

To demonstrate the superiority of our proposed solution, we compare our method with the following baselines:

CF [45]: Collaborative filtering algorithm is an algorithm that uses the preferences of a group of similar interests and common experience to recommend information that users are interested in.

FISM [46]: This is an item-to-item collaborative filtering algorithm, but no attention mechanism is used to distinguish the weight of historical data.

NAIS [44]: This is an item-to-item collaborative filtering algorithm, which uses an attention mechanism to distinguish the weight of historical data. Used as a basic recommendation model in this paper.

light-GCN [47]: This algorithm learns user and project embedding by linearly propagating user and project embedding on the user-project interaction graph, and uses the weighted sum of the embedding learned at all layers as the final embedding.

HRL [8]: Is an algorithm that uses the basic recommendation model and the hierarchical reinforcement model for joint training.

In experiments, we use the Hit Ratio of top K items (HR@K) and Normalized Discounted Cumulative Gain of top K items (NDCG@K) [8] as our evaluations metrics. To be more specific, HR@K is an index based on recall, which is used to measure the percentage of instances successfully recommended in top-K, and NDCG@K is an index based on

accuracy, which illustrates the predicted position of the instance. In this article, we set K to 5 and 10, calculate all indicators including 1 positive instance and 99 negative instances, and get the average score of all user sequences.

5. Experimental Results and Analysis

In this section, we list the experimental results of our model and baseline methods, then analyze the experimental results, including the analysis of comparative experiments and the analysis of each component of the model.

5.1. Experimental Results

The experimental results are shown in Tables 3 and 4, from which we have the following observations:

1. For school borrowing data, it can be seen from Table 3 that is in prediction performance, our model is better than the baseline methods, compared with the collaborative filtering algorithm, and our algorithm uses feature vectors to classify books, which can better predict what kind of books students like and recommend these books to students. Collaborative filtering algorithms cannot classify books, so it is difficult to recommend sparse data. For goodbooks data, the recommendation results of our model are significantly better than those of other models.
2. The two algorithms of FISM and NAIS are item-to-item collaborative filtering algorithms. They use a deep learning method, and the NAIS algorithm adds an attention mechanism to the FISM algorithm, but sparse data will have a greater impact on the training process of these two algorithms, resulting in poorer final results. The results obtained by the light-GCN algorithm are slightly better, but not particularly perfect.
3. The HRL algorithm combines reinforcement learning with deep learning removes noise in the sequence and improves the model's ability to process sparse data to a certain extent. Our CHRL algorithm deals with both sparse data and noise, and the results are better than other algorithms. In Goodbooks data, the HRL algorithms cannot get a good result, and our improved model CHRL can get better results than other models, which proves the effectiveness of our model.

Table 3. Recommendation performance on school borrowing data.

	HR@5	**HR@10**	**NDCG@5**	**NDCG@10**
CF	0.4518	0.4877	0.2856	0.2736
FISM	0.2358	0.3238	0.1771	0.2052
NAIS	0.2149	0.2860	0.1599	0.1828
Light-GCN	0.4702	0.5890	0.3231	0.3768
HRL-NAIS	0.6509	0.7834	0.4722	0.5156
CHRL	0.8293	0.9212	0.5909	0.6213

Table 4. Recommendation performance on Goodbooks data.

	HR@5	**HR@10**	**NDCG@5**	**NDCG@10**
CF	0.4070	0.5749	0.1004	0.1214
FISM	0.4017	0.5442	0.2791	0.3251
NAIS	0.3546	0.4988	0.2429	0.2894
Light-GCN	0.4294	0.5925	0.2821	0.3460
HRL-NAIS	0.2155	0.3216	0.1646	0.1927
CHRL	0.4811	0.7013	0.3682	0.4388

5.2. Model Analysis

Analysis of clustering component. The main purpose of using NAIS in the basic recommender is to analyze the characteristics of books and classify them more accurately. From the experimental results, it can be seen that if only NAIS is used for the book

recommendation, the recommendation results will be poor. The reason for this situation is that there are many noises in the sequence, and it is difficult to use the model to accurately locate the user's interest. However, the data can be accurately classified after NAIS processing, so that the clustering model can better process the book information. So using NAIS as a basic recommender is an important step in CHRL.

For book data with sparse data, HRL is not stable enough. We used library data to perform ten experiments on HRL and CHRL. Table 5 shows the average and standard deviation of the ten results. In ten experiments, HRL performed good results only twice, and the average of the ten experiments also performed poorly. This is because the reinforcement learning model deleted too much data in the sparse data sequence, resulting in too little data in the sequence. The model cannot correctly predict the next item in the sparse data sequence. As can be seen, our model in ten experiments has shown better results, indicating that our model on stability is also excellent. This is because clustering sparse data can make reinforcement learning more stable. This result proves the effectiveness of the clustering component in our model.

Table 5. The average and standard deviation of HRL and CHRL.

		HR@5	HR@10	NDCG@5	NDCG@10
HRL	average	0.2482	0.3292	0.1785	0.2045
	standard deviation	0.2043	0.2374	0.1473	0.1581
CHRL	average	0.7777	0.9020	0.5372	0.5782
	standard deviation	0.0373	0.0133	0.0538	0.0430

Analysis of sequence modification component. From the comparison of the experimental results obtained by NAIS and our model, it can be seen that the use of hierarchical reinforcement learning in the book recommendation is effective. Hierarchical reinforcement learning can modify the student sequence to remove the noise in the sequence. This step is very important in the book recommendation with sparse data. Table 6 shows the recommended results after the basic model training, the sequence modification component, and the joint training. It can be seen that the sequence modification component has greatly improved the recommendation results. This is because the clustering module accurately classifies the book categories, and hierarchical reinforcement learning can well grasp the user's interest, and can accurately find the books that the user may like among many books.

Table 6. The recommended results after the basic model training, the sequence modification component, and the joint training.

	HR@5	HR@10	NDCG@5	NDCG@10
Basic recommender	0.2149	0.2860	0.2860	0.1828
Sequence modification component	0.7583	0.8208	0.5527	0.5826
Jointly training	0.8293	0.9212	0.5909	0.6213

Analysis of joint training component. The last step of the model is joint training. The joint training feeds the modified sequence to the basic model for retraining, and then clusters the results of the basic model training and feeds it to hierarchical reinforcement learning. Hierarchical reinforcement learning modifies user borrowing sequence. Jointly training is such a cyclical process. After continuous training, the model got the final result. From the results, it can be seen that joint training is also an indispensable part of the entire model.

5.3. Analysis of Hyperparameters

For the analysis of hyperparameters, we focused on the number of categories in the clustering component and the learning rate in each component.

Categories in the clustering component. The hyperparameters of the clustering model are the number of categories. If the number of categories is set too low, many books with little relevance will be classified into one category, and the hierarchical reinforcement learning model will have difficulty distinguishing these books, which will affect the final result. If the number of categories is set too high, the classification ability of the clustering model will be weak, and the hierarchical enhancement model will delete too many items, which will easily lead to poor final results. We have done many parameter experiments, the results of the experiment are shown in the Table 7. The data in the Table 7 proves this conclusion. We finally set the number of categories in the school library data to 2000, and the number of categories in the goodbooks data is set to 1000. This number of categories will get a good result.

Table 7. The result of different values of the category in the school borrowing data.

Category	HR@5	HR@10	NDCG@5	NDCG@10
10,000	0.3460	0.4057	0.2351	0.2576
5000	0.6528	0.7021	0.4953	0.5182
2000	0.8293	0.9212	0.5909	0.6213
1000	0.8053	0.8974	0.5680	0.5926

Learning rate. In our model, both the basic model and the sequence modification module have learning rate settings. In the basic model, through multiple experiments on the basic model, we set the learning rate to 0.02 to get the fastest training efficiency. In the sequence modification component, we tested multiple data and found that a high learning rate would cause the model to be unstable, and a low learning rate will cause the model to train too slowly. Finally, we set the learning rate of the sequence modification component to 0.05. This learning rate can train this module stably and quickly.

6. Conclusions and Future Work

In this work, our purpose is to solve the problem of data sparseness and noise in book recommendations in the school digital library environment, so we proposed a hierarchical reinforcement learning-based method for the book recommendation task to solve the encountered data noise and data sparsity challenges. More specifically, we used clustering to classify the data and effectively solved the problem of sparse data in the school library environment. From the experimental results, we can observe that our model has a significant improvement over other related methods. The performance of our model on HR@10 and NDCG@10 have an improvement of more than 10% over other compared methods, which demonstrates the effectiveness of reinforcement learning-based recommendation method.

In the actual operation of the university digital library, it is necessary to input the existing data into the model for training, and then the model will recommend the corresponding books to the user based on the user's information. After the user selects a book according to his own preferences, the model will recommend the user the next time according to the user's new sequence. Since our model uses the borrowing sequence data and clustering model, when our model is used in areas other than book recommendation, the data needs to have time information and category information.

We will try to use this model in other areas of the school. For example, we will use our model to recommend a course selected by a student when the student selects a course, or use this model to recommend restaurant meals in a student restaurant. In the future, we will study the recommendation of combining library data with other school data, such as library data and course selection data or add knowledge graph data of library books to the model to make recommendations more accurate.

Author Contributions: Conceptualization, Y.W.; Data curation, W.L.; Funding acquisition, L.G., X.W.; Project administration, X.W.; Software, Y.W.; Supervision, L.G.; Validation, L.X., F.L.; Visualization, B.G.; Writing—original draft, Y.W.; Writing—review & editing, L.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Natural Science Foundation of China (No. 61602282), China Postdoctoral Science Foundation (No. 2016M602181), and Social Sciences Foundation of Shandong (No. 19CTQJ06, Research on Library Personalized Recommendation Service Based on User Portrait).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available on request due to privacy. The data presented in this study are available on request from the corresponding author. The data are not publicly available due to part of the data is private.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Goldberg, D.; Nichols, D.; Oki, B.M.; Terry, D. Using collaborative filtering to weave an information tapestry. *Commun. ACM* **1992**, *35*, 61–70. [CrossRef]
2. Pazzani, M.J.; Billsus, D. Content-based recommendation systems. In *The Adaptive Web*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 325–341.
3. Zhang, Z.K.; Zhou, T.; Zhang, Y.C. Tag-aware recommender systems: A state-of-the-art survey. *J. Comput. Sci. Technol.* **2011**, *26*, 767. [CrossRef]
4. Burke, R. Knowledge-based recommender systems. *Encycl. Libr. Inf. Syst.* **2000**, *69*, 175–186.
5. Yang, S.T.; Hung, M.C. A model for book inquiry history analysis and book-acquisition recommendation of libraries. *Libr. Collect. Acquis. Tech. Serv.* **2012**, *36*, 127–142. [CrossRef]
6. Sohail, S.S.; Siddiqui, J.; Ali, R. An OWA-based ranking approach for university books recommendation. *Int. J. Intell. Syst.* **2018**, *33*, 396–416. [CrossRef]
7. Priyanka, K.; Tewari, A.S.; Barman, A.G. Personalised book recommendation system based on opinion mining technique. In Proceedings of the 2015 Global Conference on Communication Technologies (GCCT), Thuckalay, India, 23–24 April 2015; pp. 285–289.
8. Zhang, J.; Hao, B.; Chen, B.; Li, C.; Chen, H.; Sun, J. Hierarchical Reinforcement Learning for Course Recommendation in MOOCs. *AAAI Conf. Artif. Intell.* **2019**, *33*, 435–442. [CrossRef]
9. Ansari, A.; Essegaier, S.; Kohli, R. Internet Recommendation Systems. *J. Market. Res.* **2000**, *37*, 363–375. [CrossRef]
10. Ziegler, C.N.; McNee, S.M.; Konstan, J.A.; Lausen, G. Improving recommendation lists through topic diversification. In Proceedings of the 14th International Conference on World Wide Web, Chiba, Japan, 10–14 May 2005; pp. 22–32.
11. Konstas, I.; Stathopoulos, V.; Jose, J.M. On social networks and collaborative recommendation. In Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Boston, MA, USA, 19–23 July 2009; pp. 195–202.
12. Robillard, M.; Walker, R.; Zimmermann, T. Recommendation systems for software engineering. *IEEE Softw.* **2009**, *27*, 80–86. [CrossRef]
13. Smyth, B. Case-based recommendation. In *The Adaptive Web*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 342–376.
14. Fu, X.; Budzik, J.; Hammond, K.J. Mining navigation history for recommendation. In Proceedings of the 5th International Conference on Intelligent User Interfaces, New Orleans, LA, USA, 9–12 January 2000; pp. 106–112.
15. Drineas, P.; Kerenidis, I.; Raghavan, P. Competitive recommendation systems. In Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing, Montreal, QC, Canada, 19–21 May 2002; pp. 82–90.
16. Sabitha, S.; Choudhury, T. Proposed approach for book recommendation based on user k-NN. In *Advances in Computer and Computational Sciences*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 543–558.
17. Goel, A.; Khandelwal, D.; Mundhra, J.; Tiwari, R. Intelligent and integrated book recommendation and best price identifier system using machine learning. In *Intelligent Engineering Informatics*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 397–412.
18. Mikawa, M.; Izumi, S.; Tanaka, K. Book recommendation signage system using silhouette-based gait classification. In Proceedings of the 2011 10th International Conference on Machine Learning and Applications and Workshops, Honolulu, HI, USA, 18–21 December 2011; Volume 1, pp. 416–419.
19. Xin, L.; Haihong, E.; Junde, S.; Meina, S.; Junjie, T. Collaborative book recommendation based on readers' borrowing records. In Proceedings of the 2013 International Conference on Advanced Cloud and Big Data, Nanjing, China, 13–15 December 2013; pp. 159–163.

20. Maneewongvatana, S.; Maneewongvatana, S. A recommendation model for personalized book lists. In Proceedings of the 2010 10th International Symposium on Communications and Information Technologies, Tokyo, Japan, 26–29 October 2010; pp. 389–394.
21. Yang, S.T. An active recommendation approach to improve book-acquisition process. *Int. J. Electron. Bus. Manag.* **2012**, *10*, 163–173.
22. Tewari, A.S.; Kumar, A.; Barman, A.G. Book recommendation system based on combine features of content based filtering, collaborative filtering and association rule mining. In Proceedings of the 2014 IEEE International Advance Computing Conference (IACC), Gurgaon, India, 21–22 February 2014; pp. 500–503.
23. Sohail, S.S.; Siddiqui, J.; Ali, R. Book recommendation system using opinion mining technique. In Proceedings of the 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Mysore, India, 22–25 August 2013; pp. 1609–1614.
24. Kanetkar, S.; Nayak, A.; Swamy, S.; Bhatia, G. Web-based personalized hybrid book recommendation system. In Proceedings of the 2014 International Conference on Advances in Engineering & Technology Research (ICAETR-2014), Unnao, India, 1–2 August 2014; pp. 1–5.
25. Vaz, P.C.; Martins de Matos, D.; Martins, B.; Calado, P. Improving a hybrid literary book recommendation system through author ranking. In Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries, Washington, DC, USA, 10–14 June 2012; pp. 387–388.
26. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv* **2013**, arXiv:1312.5602.
27. Hausknecht, M.; Stone, P. Deep recurrent q-learning for partially observable mdps. *arXiv* **2015**, arXiv:1507.06527.
28. Wang, Z.; Schaul, T.; Hessel, M.; Van Hasselt, H.; Lanctot, M.; De Freitas, N. Dueling network architectures for deep reinforcement learning. *arXiv* **2015**, arXiv:1511.06581.
29. Van Hasselt, H.; Guez, A.; Silver, D. Deep reinforcement learning with double q-learning. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016.
30. Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized experience replay. *arXiv* **2015**, arXiv:1511.05952.
31. Hessel, M.; Modayil, J.; Van Hasselt, H.; Schaul, T.; Ostrovski, G.; Dabney, W.; Horgan, D.; Piot, B.; Azar, M.; Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
32. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; pp. 1928–1937.
33. Vezhnevets, A.; Mnih, V.; Osindero, S.; Graves, A.; Vinyals, O.; Agapiou, J. Strategic attentive writer for learning macro-actions. *arXiv* **2016**, arXiv:1606.04695.
34. Vezhnevets, A.S.; Osindero, S.; Schaul, T.; Heess, N.; Jaderberg, M.; Silver, D.; Kavukcuoglu, K. Feudal networks for hierarchical reinforcement learning. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; Volume 70, pp. 3540–3549.
35. Nachum, O.; Gu, S.S.; Lee, H.; Levine, S. Data-efficient hierarchical reinforcement learning. *arXiv* **2018**, arXiv:1805.08296.
36. Theocharous, G.; Thomas, P.S.; Ghavamzadeh, M. Personalized ad recommendation systems for life-time value optimization with guarantees. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015.
37. Wang, X.; Wang, Y.; Hsu, D.; Wang, Y. Exploration in interactive personalized music recommendation: A reinforcement learning approach. *ACM Trans. Multimed. Comput. Commun. Appl. TOMM* **2014**, *11*, 1–22. [CrossRef]
38. Zheng, G.; Zhang, F.; Zheng, Z.; Xiang, Y.; Yuan, N.J.; Xie, X.; Li, Z. DRN: A deep reinforcement learning framework for news recommendation. In Proceedings of the 2018 World Wide Web Conference, Lyon, France, 23–27 April 2018; pp. 167–176.
39. Wang, L.; Zhang, W.; He, X.; Zha, H. Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 2447–2456.
40. Zhao, X.; Zhang, L.; Ding, Z.; Xia, L.; Tang, J.; Yin, D. Recommendations with negative feedback via pairwise deep reinforcement learning. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1040–1048.
41. Zhao, X.; Xia, L.; Zhang, L.; Ding, Z.; Yin, D.; Tang, J. Deep reinforcement learning for page-wise recommendations. In Proceedings of the 12th ACM Conference on Recommender Systems, Vancouver, BC, Canada, 2–7 October 2018; pp. 95–103.
42. Chen, S.Y.; Yu, Y.; Da, Q.; Tan, J.; Huang, H.K.; Tang, H.H. Stabilizing reinforcement learning in dynamic environment with application to online recommendation. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1187–1196.
43. Rohde, D.; Bonner, S.; Dunlop, T.; Vasile, F.; Karatzoglou, A. Recogym: A reinforcement learning environment for the problem of product recommendation in online advertising. *arXiv* **2018**, arXiv:1808.00720.
44. He, X.; He, Z.; Song, J.; Liu, Z.; Jiang, Y.G.; Chua, T.S. Nais: Neural attentive item similarity model for recommendation. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 2354–2366. [CrossRef]

45. Linden, G.; Smith, B.; York, J. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet Comput.* **2003**, *7*, 76–80. [CrossRef]
46. Kabbur, S.; Ning, X.; Karypis, G. Fism: Factored item similarity models for top-n recommender systems. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL, USA, 11–14 August 2013; pp. 659–667.
47. He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; Wang, M. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. *arXiv* **2020**, arXiv:2002.02126.