Scientific Research Publishing

Open Journal of Applied Sciences, 2022, 12, 134-154
https://www.scirp.org/journal/ojapps
ISSN Online: 2165-3925
ISSN Print: 2165-3917

A Survey of Online Course Recommendation Techniques

Jinliang Lu

Jinan University, Guangzhou, China
Email: jinliang@stu2019.jnu.edu.cn

How to cite this paper: Lu, J.L. (2022) A Survey of Online Course Recommendation Techniques. *Open Journal of Applied Sciences*, **12**, 134-154.
https://doi.org/10.4236/ojapps.2022.121010

Received: December 27, 2021
Accepted: January 22, 2022
Published: January 25, 2022

Abstract

With the development of information technology, online learning has gradually become an indispensable way of knowledge acquisition. However, with the increasing amount of data information, it is increasingly difficult for people to find appropriate learning materials from a large number of educational resources. The recommender system has been widely used in various Internet applications due to its high efficiency in filtering information, helping users to quickly find personalized resources from thousands of information, thereby alleviating the problem of information overload. In addition, due to its great use value, many new researches have been proposed in the field of recommender systems in recent years, but there are not many works on online course recommendation at present. Therefore, this paper aims to sort out the existing cutting-edge recommendation algorithms and the work related to online course recommendation, so as to provide a comprehensive overview of the online course recommender system. Specifically, we will first introduce the main technologies and representative work used in the online course recommender system, explain the advantages and disadvantages of various technologies, and finally discuss the future research direction of the online course recommender system.

Keywords

Information Overload, Recommender Systems, Personalization, Online Course

1. Introduction

In recent years, the emergence of online education platforms and massive open online courses (MOOCs) has attracted widespread interest. The establishment of various MOOCs platforms, including XuetangX, Chinese University MOOC,

and Coursera, provides convenient education for more than 100 million users around the world, and provides a low-cost opportunity to access excellent courses in many top universities [1]. Due to its convenience and abundance of teaching resources, online learning has gradually become a common way of learning. Especially due to the influence of COVID-19 in recent two years, traditional teaching methods have become difficult to implement in many places. Online learning plays an important role at this time. The rapid development of online learning has changed the traditional teaching mode and made people study anytime and anywhere a reality. Online learning has become an important way for people to learn knowledge, expand their skills, and conduct academic research [2].

However, while online learning brings many conveniences, it also leads to the increasingly serious problem of information overload [3]. Due to the rapid growth of the number of educational resources, it has gradually become difficult for people to choose suitable courses for learning. In order to effectively solve the information overload, people first use search engine tools, which take keyword search as the core. With the help of search engines, users can search relevant courses simply by typing in the keywords of the courses they want to query. Due to its convenient and easy-to-use features, search engines reduce the problem of information overload to a certain extent. However, in online learning scenarios, users may not have a clear purpose. In addition, the selection of courses usually requires some pre-knowledge, and users usually lack an in-depth understanding of the overall knowledge structure, and do not know which courses are suitable for them, so they cannot determine their course selection goals. In the absence of a definite purpose, the search engine cannot provide effective information to users due to their self-limiting nature.

The recommender system is another effective means to solve the information overload. Given the historical interaction data of users and items, the recommender system can effectively capture hidden information such as the user's personalized preferences and item attributes based on these data, so as to obtain information from appropriate materials filtered out from the massive resources and displayed to users [4]. At present, recommendation systems have been widely used in the fields of e-commerce and social media platforms, and have become an important part of many portals, playing an increasingly important role [5].

The use of the recommender system effectively solves the problem of information overload. For the course recommender system, another problem needs to be solved, that is, how to achieve personalized recommendation. Traditional recommender systems usually only consider the items that the user has interacted with, and then recommend the next item that the user may be interested in, without fully considering the user's personalized preferences. In the online learning scenario, in addition to extracting the user's preferences based on the user's historical course selection records and their performance in these courses, it is also necessary to take the user's knowledge structure into consideration

based on the knowledge dependence between courses. In addition, it is also necessary to consider the user's career planning and other factors, so as to help users select suitable courses from hundreds of courses for learning. In this way, it can provide personalized recommendations while avoiding the common cold start problem in the recommendation system [6] [7]. For an efficient online course recommender system, the recommender system should be able to capture users' interests according to users' historical course selection records, extract users' personalized characteristics, and comprehensively consider users' majors, knowledge structure, career planning and other factors to make accurate and personalized recommendations for users.

Over the last decade, researchers have done a lot of work on recommender systems. However, there are still few relevant applications of recommendation algorithms in online course recommendation at present. How to apply the current state-of-the-art recommendation algorithm to online course recommendation is the main challenge faced by the current online course recommendation system. In order to effectively promote the research on online course recommendation, it is necessary to refer to the existing cutting-edge recommendation algorithms. Compared with other e-commerce scenarios or social media recommendation scenarios, online learning scenarios have many differences, and the recommendation for online courses has important research value for online learning scenarios [8]. Therefore, sorting out, analyzing and summarizing existing recommendation algorithms plays a very important role in the further study of online course recommender system in the future.

The goal of this article is to comprehensively review the current cutting-edge recommendation algorithms and the latest work on online course recommendation. Researchers and practitioners interested in the online course recommender system can have a holistic understanding of the recommender system, and quickly learn about the latest developments in the current online course recommender system, and learn how the online course recommender system recommends courses to users. The key contributions of this paper are as follows:

- We propose a new classification system for online course recommender system, which divides recommendation algorithms into general recommendation and sequential recommendation according to whether the order of course selection is taken into account, and distinguishes recommendation algorithms according to the technology used, reorganizing the existing course recommender system model.
- According to the new classification system of online course recommendation algorithm, we sorted out the problems to be solved under different technologies, introduced the most representative work at present, and summarized the overall ideas of each recommendation model to explain how they solve the problems.
- We discuss the limitations and challenges of various recommendation algorithms, and point out the shortcomings of current online course recommender system research and the possible future development direction.

The remaining of this article is organized as follows: The Section 2 is an overview of online course recommender system, introducing the basic concepts and framework of the recommendation system. In Section 3, we will introduce the general recommendation model, mainly introducing the evolution process of the general recommendation method, including from the classic collaborative filtering method to the neural network-based model and the recently popular graph-based neural network model Section 4 introduces the sequence recommendation model. Different from the general recommendation model, the sequence recommendation uses algorithms commonly used in natural language processing such as recurrent neural networks and self-attention mechanisms to take into account the interaction sequence of the user's historical interaction items to improve recommendation effects. We will show how they are applied to the recommender system. In Section 5, we will discuss the datasets and evaluation metrics commonly used in the current online course recommender system. Finally, we summarize the online course recommendation algorithm, point out the shortcomings and challenges of the current research, and discuss the promising future research direction.

2. Overview of Course Recommender System

Before we dive into the details of this survey, we start with a brief introduction to the basic concepts regarding online course recommender systems.

Based on the attributes of users and items or the historical interaction data between them, the recommender system can infer users' personalized preferences and hidden features of items, and further recommend items that users may be interested in. Therefore, a typical recommender system task is generally defined as generating a list of recommended items for users in a specific recommendation environment or predicting users' preference for a specific item. As for the course recommender system, similarly, it is mainly used to recommend a course list for the user or predict the user's preference for a certain course, so as to predict the course that the user may attend next.

The input data of the recommender system usually includes explicit data (ratings, likes/dislikes) and implicit data (clicking, browsing and other behavioral data), or user portraits (gender, age, etc.) and item content (text, image, and other description or content), or label, comment and other side information. For versatility and ease of practice, this article only considers the interaction records between users and items. Such a recommender system is easier to transplant to course recommendation. In the course recommender system, we only need to pay attention to the user's historical course selection record information. Therefore, the item mentioned in this article usually refers to courses, and the interaction record between the user and the item refers to the record of the user selecting courses. Adding side information such as course content into consideration may improve the recommendation effect. However, due to the fact that there are still relatively few related researches and the complexity of the research is in-

creased, this article will not discuss it. It is worthy of further research in the future.

After the user's course selection record data is collected, the data needs to be input into the model. How to represent these data in the recommender system is the first concern. In recommender systems, data is usually represented in the form of vectors. For example, using one-hot encoding to represent users and items is the easiest way. Assuming there are 3 users and 4 items, user 2 is represented by [0, 1, 0], and item 3 is represented by [0, 0, 1, 0]. By inputting the user's course selection data into the recommendation model, different user and item representations can be obtained. This kind of representation is also called latent representation, which can be used to input into other models for further processing, or to generate a recommendation list. The whole processing framework is shown in Figure 1.

In the model layer, a wide range of models are available, including autoencoders, neural networks, reinforcement learning, and so on. In the output layer, using the learned latent representation of users and items, the recommended list of items can be generated by using inner product, SoftMax, similarity calculation and other methods. This article mainly focuses on the advanced technologies and algorithms used in online course recommendation, that is, the model level. According to whether to consider the order of historical course selection records, online course recommendation systems can generally be divided into general recommendation tasks and sequential recommendation tasks.

3. General Recommendation Method

General recommendation assumes that users have static preferences. Based on historical data of interaction between users and items, the similarity between users and items is modeled based on explicit (e.g., score) or implicit (e.g., click) feedback. In general recommendation, the most representative method is collaborative filtering, which is one of the most widely used and oldest algorithms in

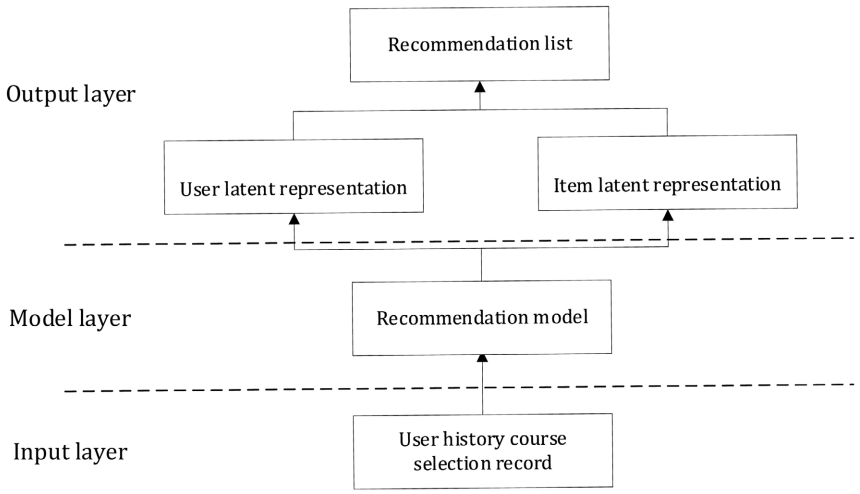

Figure 1. Course recommendation system framework.

recommender systems [9] [10]. The recommendation algorithm based on collaborative filtering models the user-item historical interaction record data to predict the user's preference or rating of the item, and then recommends the top-N items that may be of interest to the user, as shown in **Figure 2** [11].

The basic idea of collaborative filtering is to assume that similar users have similar preferences. Therefore, other users similar to the target user can be identified, and then the items liked by these similar users can be recommended to the target user. At present, there are many researches on collaborative filtering. For example, some early memory-based models realized user (item) based collaborative filtering by calculating the similarity of users (items). Recommendation based on collaborative filtering usually expresses user-item interaction in the form of a matrix, and then turns the recommendation task into a matrix filling task [12].

Suppose we have M users ($U = \{u_1, \cdots, u_M\}$) and N items ($I = \{i_1, \cdots, i_N\}$), the interaction between users and projects can be expressed in the form of a matrix, usually denoted by $R \in \mathbb{R}^{M \times N}$. For explicit interaction, the interaction matrix is also called scoring matrix. The rows of the interaction matrix represent users and the columns represent items. We denote row u and column i of the matrix as r_{ui}, which represents user u's preference score for item i. Matrix factorization algorithm, by decomposing the user item interaction matrix into low-dimensional hidden factor vectors as the representation of users and items, and then using the trained representation to reconstruct the interaction matrix $\hat{R}$, where $\hat{r}_{ui}$ represents the predicted score of user u for item i, therefore $\hat{R}$ is also called the prediction matrix. Matrix factorization is one of the most typical collaborative filtering methods by reconstructing the interaction matrix to predict whether the user is interested in items that have not yet interacted, so as to further recommend the items that may be of interest to the target user and complete the recommendation task [12] [13]. Matrix factorization is widely used in various applications due to its simplicity. However, in the face of an increasing number of users and items, matrix factorization in practical applications faces a serious data sparse problem. There may be very few items that have interacted with users, resulting in poor performance.

As can be seen from the example of matrix factorization, collaborative filtering first obtains the vector representation of users and items in the recommendation

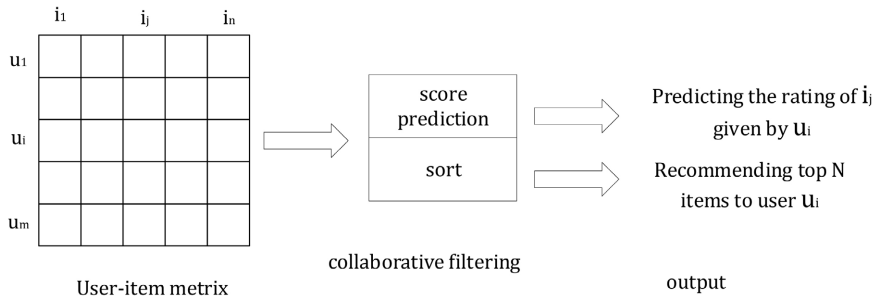

Figure 2. The recommendation algorithm based on collaborative filtering.

system. By encoding the user ID and item ID into a one-hot encoding, and then inputting them into the embedding layer (hidden factor model), the vector representation of the user and the item can be obtained, denoted by p_u and q_i respectively. This process is also called representation learning. Based on the learned representation, the interaction between the user and the item can be re-established, and the user u's preference prediction for item i can be obtained, denoted as $\hat{r}_{ui}$. This process is also called interaction modeling. For example, it can be constructed with the inner product of p_u and q_i, that is, $\hat{r}_{ui} = p_u^T q_i = \sum_{f=1}^{k} p_{uf} q_{if}$, where k is the dimension of hidden space. The inner product can effectively express the similarity between the user's items. The greater the value of the $\hat{r}_{ui}$, the more likely the user will like the item. Because of its simplicity and efficiency, many works have used this method and achieved good results. Many of the current work can be classified as improvements to representation learning or interaction construction.

3.1. Neural Network-Based Method

In recent years, deep neural network technology has emerged. Due to its excellent nonlinear modeling ability, it has strong expressive ability. Deep learning has been widely used in various artificial intelligence applications such as computer vision and natural language processing [14] [15] [16]. Naturally, how to introduce deep learning into the recommender system has been a research hotspot in recent years.

For instant, a multilayer neural network can be used to fit the inner product operation in the interaction construction function of matrix factorization. Most previous models use inner products to model interactions between users and items, such as the matrix factorization model described above. Since only linear modeling is used, matrix factorization can be regarded as a linear model with latent factors. However, since it only model linear interactions, it may not be able to effectively capture the complex relationship between users and items, resulting in a suboptimal model [17]. According to the characteristic that neural network can fit any complex continuous function, researchers try to use multi-layer neural network to fit the inner product operation of matrix factorization interaction function, and proposed the NCF model [18]: $\hat{r}_{ui} = f_{MLP}\left(p_{u\|}q_{if}\right)$. Due to the nonlinear modeling characteristics of neural networks, NCF has better expressive capabilities than matrix factorization models. In addition, the author of the NCF model proposed that combining the linearity of the matrix factorization and the nonlinearity of the neural network model can further improve the recommendation effect of the model.

Referring to the idea of autoencoder, AutoRec [19] and CDAE [20] use neural networks to implement encoders and decoders to reconstruct the user item interaction matrix. Taking one row (user-based) or one column (item-based) of the interaction matrix R as input, and projecting the input features to a low-dimensional vector space using a neural network, a hidden layer representation of

the user or item can be obtained (encoding process). Then use the neural network representation to project the hidden layer representation to the normal space, decode it to restore the input features, and the output result is the user's preference prediction (decoding process), as shown in Figure 3. Utilizing the characteristics that the target value and input value are similar in the autoencoder, the model parameters are optimized by minimizing the reconstruction error, so as to reconstruct the ratings of items for which the user has not interacted. Different from AutoRec, in the encoding process, the CDAE model considers the personalization factors of different users, and adds a user factor for each user, making the model more semantic. In order to make the model more robust, the CDAE model performs noise processing on the input features (implemented by dropout or adding Gaussian noise).

In addition to directly using the user's ID to establish a vector representation of the user, some methods propose to use the representation of the user's historical interactive items to construct a better user representation. For example, a simple idea is to directly use the average representation value of all historical interaction items of the user as the user's representation, as implemented by FISM [21]:

$$\hat{r}_{ui} = \left(\frac{1}{|\mathcal{R}_u|^{\alpha}} \sum_{j \in \mathcal{R}_u} q'_j \right)^T q_i \tag{1}$$

where $\mathcal{R}_u$ represents the item that has interacted with user u, and α is the hyperparameter that controls the regularization. Similarly, SVD++ [22] adds the user representation constructed by ID and the user representation constructed by historical interactive items, and uses it as a new user representation. Although these methods have achieved good performance, in fact, different items have different contributions to user preferences, and equal aggregation of user history items will limit their ability to express.

Obviously, different weights should be given to different items to obtain better user representation, higher weights should be given to items that can better reflect user interests, and lower weights should be given to items that are less or even irrelevant to user interests. The use of attention mechanism can well solve

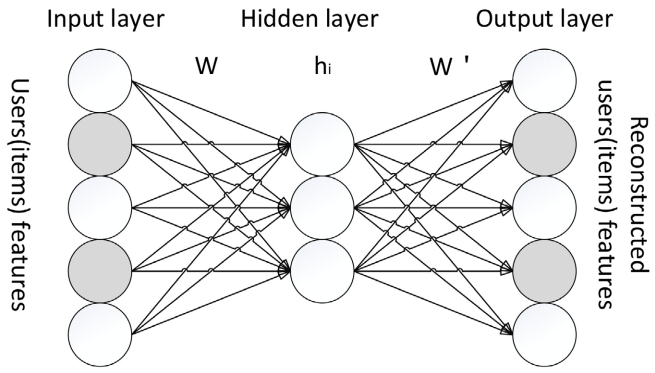

Figure 3. Autoencoder.

this problem. For example, NAIS [23] model gives different weights to user historical interaction items according to different target items:

$$\hat{r}_{ui} = \left(\sum_{j \in \mathcal{R}_u} a_{ij} q'_j \right)^T q_i$$
$$a_{ij} = \frac{\exp\left(\mathcal{F}\left(q_i, q'_j\right)\right)}{\left[\sum_{k \in \mathcal{R}_u} \exp\left(\mathcal{F}\left(q_i, q'_k\right)\right)\right]^\beta} \quad (2)$$

where a_{ij} is the attention weight, which represents the contribution of the historical interaction item j to the user's preference for the target item i. $\mathcal{F}(\cdot,\cdot)$ can be an MLP operation or a simple inner product function. The value range of β is [0, 1], which is a hyperparameter used to smooth the length of different historical interactive items of users. By using the attention mechanism, the representation ability of the model can be effectively enhanced, and the prediction effect can be improved. Similar work that uses the attention mechanism also includes Attentive Collaborative Filtering (ACF) [24], Deep Item-based CF model (DeepICF) [25] and Deep Interest Network (DIN) [26], etc.

3.2. Graph Neural Network-Based Method

In recent years, Graph Neural Network (GNN) technology has been widely used in many fields due to its outstanding performance in graph structure data learning [27]. In the recommender system, most of the information essentially has a graph structure. For example, user item interaction data can be represented using a bipartite graph structure, as shown in **Figure 4**, which is more intuitive than using a matrix to represent the interaction between users and items. By using the graph structure, graph algorithms can be naturally applied to model user item interaction data. Since GNN has obvious advantages for representation learning and can effectively model the bipartite graphs of user items, there have been many attempts to apply graph algorithms to recommender systems.

By observing the bipartite graph structure, it can be found that the user's historical interaction items are actually the first-order neighbors of the user node in the graph. The traditional deep learning method only models the user's historical interaction data, that is, only considers the first-order neighbors in the user

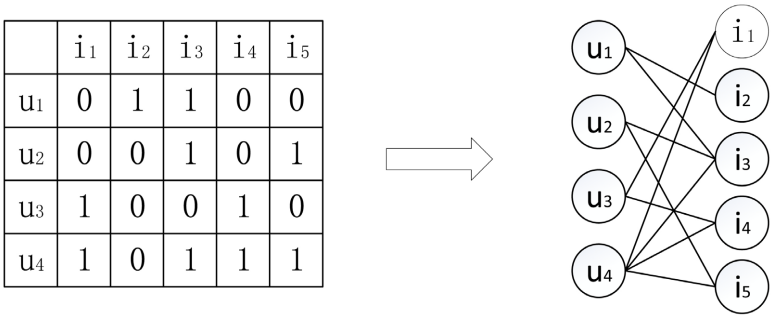

Figure 4. User-item bipartite graph.

interaction data, and does not consider its second-order neighbors (other users who have the same interactive items as the user) and other high-order neighbors. Therefore, a natural extension is to encode the high-order interaction information between users and items into their representation, and model the high-order connectivity in the graph structure to improve the expression ability of the model.

The GNN-based recommender system first uses the user item representations P^0 and Q^0 constructed by ID as 0-order representations, and then iteratively transmits and updates the node information on the graph through the graph neural network layer. The $(l+1)^{\text{th}}$-order representation of a user p_u^{l+1} (item p_i^{l+1}) can be constructed from its l^{th}-order item (user) neighbor representation. For example, the $(l+1)^{\text{th}}$-order representation p_u^{l+1} of user u can be calculated as:

$$\begin{aligned} p_u^{l+1} &= \mathcal{G}\left(p_u^l, n_u^l\right) \\ n_u^l &= \text{Agg}\left(q_j^l \mid j \in \mathcal{N}_u\right) \end{aligned} \tag{3}$$

where q_j^l is the representation of item j at layer l, $\mathcal{N}_u$ is represented as the item connected to user u in the bipartite graph, n_u^l is the representation obtained by aggregating the neighbor representations of user u, and $\mathcal{G}$ is graph neural network operation. By iteratively updating the nodes on the graph, after the L-layer graph neural network operation, each node in the graph will contain the information of the entire graph and the high-order connectivity between users and items. After obtaining the L layer representation, directly using the last layer as the final node representation usually cannot achieve the best results, and the increase in the number of layers is likely to cause the problem of over-smoothness [28]. Moreover, different layers can capture different semantic information. For example, the first layer constructs direct interaction information between users and items, and the second layer captures users or items with overlapping interactions. Blindly using the nodes of the last layer as the final representation may lose a lot of useful information. Therefore, NGCF [29] concatenates the node representations of all layers as the final representation, and LightGCN [30] proposes to use the weighted summation or average value of the representations of all layer nodes to obtain the final representation.

4. Sequential Models

In practical application scenarios, user preferences are usually not static, but change over time. For example, in a course recommendation scenario, users often change their learning direction over time. Different from modeling the user's static preferences in general recommendation, sequence recommendation captures the sequence pattern between consecutive items, so as to recommend the user the next item that may be of interest to the user [31].

4.1. Markov Chain-Based Method

Some work assumes that the items that users recently interact with reflect their

dynamic preferences [32] [33], and adopts Markov Chain (MC) model to capture the transition information from item to item. The Markov chain-based method calculates the transition probability based on the transition matrix, and predicts the user interaction at the next time through the interaction at the previous moment. FPMC [33] combines matrix factorization and MC method to integrate the static and dynamic preferences of users.

However, because the Markov chain-based method only considers one or several recent interactions in prediction, it can only capture short-term dependencies and ignore the long-term dependencies in the sequence, which leads to its failure to achieve good results in practical applications.

4.2. Recurrent Neural Network-Based Method

Benefiting from the advantages of Recurrent Neural Network (RNN) in sequence modeling, RNN-based recommender systems use RNN units to capture the sequence patterns between user and item interactions, and model the dependencies of a given interaction sequence to predict the next possible interaction. For example, Hidasi *et al.* applied Gated Recurrent Unit (GRU), a variant of RNN, to a session-based recommender system for the first time, and proposed GRU4REC [34] and GRU4REC+ [35], respectively. By inputting the current item information in the sequence into the GRU unit, the hidden state is updated, and the next item that the user may be interested in is output according to the current hidden state. In order to effectively capture the development of item dependencies over time, GRU4REC designed session-parallel mini-batches to avoid the problem that the use of batch for accelerated training in RNN requires a unified sequence length. Apart from the use of GRU, some works [36] also applied another variant of RNN, Long Short-Term Memory (LSTM) to the recommender system. For example, Quadrana *et al.* [37] used hierarchical RNN to capture more complex dependencies in the interaction sequence. RNN brings a good effect to sequence recommendation. However, because RNN assumes that there is a certain dependency between any adjacent interactions in the interaction sequence, in actual scenarios, such an overly strong assumption usually leads to overfitting problems, generating wrong dependencies, and reducing the recommendation effect.

4.3. Convolutional Neural Networks-Based Method

Convolutional Neural Networks (CNN) are often used in image processing fields such as computer vision. Some researches [38] [39] apply CNN to sequential recommendation. Different from the RNN-based method, for a given interaction sequence, the CNN-based method composes the embedding vector of historical items in the form of a matrix, and treats the matrix as an image in time and latent space. Then, the local features of the image are obtained by learning the sequence mode, and operations such as convolution filtering are used to obtain the user's representation for subsequent recommendations. For example, Caser [38]

is a representative work, which solves the problem that only point-level sequential patterns can be modeled in Markov chain-based methods, but not union-level patterns. Moreover, because the CNN-based recommendation algorithm learns the pattern in the image region instead of directly learning the dependence between the sequences, it will not produce the problem of excessive assumptions in the RNN-based method, and to a certain extent make up for the shortcomings of the RNN-based method. However, due to the limitation of the size of the convolution kernel, the CNN-based method is affected by the size of the local receptive field. It is usually more advantageous for short-term preference modeling and cannot effectively capture long-term dependencies.

4.4. Attention Mechanism-Based Method

None of the aforementioned sequential methods pay attention to the importance of different items in the sequence. However, considering all historical interactive items equally will reduce the recommendation effect. Using the attention mechanism to integrate the item sequence and the items that have recently interacted can effectively improve the recommendation effect. For example, the author of NARM [40] found that most users' interests are concentrated at the end of the interaction sequence, but not all interaction items are related to the next interaction item. Therefore, they designed an Encoder-decoder structure. The encoder part includes Global encoder and Local encoder, both of which are composed of stacked multi-layer GRUs, which respectively model user behavior sequences and main intentions, and use the attention mechanism to learn the main intentions. Similarly, STAMP [41] designed different structures to model long-term interest and short-term interest in a session. By using RNN to model the interaction sequence to extract the user's long-term interest, the last interaction item is used to characterize the user's short-term interest, and the attention mechanism is added to improve the recommendation effect.

Due to the outstanding performance of Transformer on NLP tasks, some works use self-attention mechanisms to model sequence interactions, which adds flexibility to capturing item-to-item transition information. For example, SASRec [42] uses a self-attention mechanism to model the entire user historical interaction sequence, without any loop and convolution operations, and has achieved good results. Similarly, AttRec [43] also uses a self-attention mechanism to extract users' short-term interests, and uses metric learning to obtain users' long-term preferences to improve the recommendation effect. BERT4Rec [44] believes that the one-way structure will cause certain restrictions on the modeling of historical sequences, and the two-way Transformer structure is used for modeling to avoid the impact of the interaction sequence of high similarity items on the recommendation results.

4.5. Graph Neural Network-Based Method

When considering the interaction order of the interaction sequence, the se-

quence data can be expressed in the form of a graph structure, and then the GNN model can be used to model it. For example, the interaction sequence can be constructed in the form of a sequence graph, as shown in Figure 5.

By using GNN to model the sequence graph and learning the item transition mode in the interaction sequence, it is possible to dig out the complex interaction relationship between the user's items. Taking SRGNN [45] as an example, it first constructs a directed session graph for the interaction sequence, and uses Gated Graph Neural Network (GGNN) to learn a unified representation of all nodes in the session graph. Then, the embedding representation of the last interaction node in the sequence is taken as the user's current local interest embedding to highlight the importance of the final interaction item, and the soft-attention network is used to obtain the global embedding representation to represent the user's long-term interest. Finally, the two representations are combined by simple linear transformation to obtain hybrid embedding, and the predicted value is obtained by inner product of the final representation and candidate item representation, thus completing the recommendation task.

Similar work also includes GC-SAN [46], MA-GNN [47], etc. The GNN-based method has shown excellent recommendation effect in sequence recommendation and has become a new trend and deserves further research in the future.

4.6. Reinforcement Learning-Based Method

It can be seen from many previous works that deep learning has made great progress in many applications of recommender systems. At the same time, some recent reinforcement learning work has also achieved good results. The basic idea of reinforcement learning is to learn the optimal strategy to complete the goal by maximizing the cumulative reward value obtained by the agent from the environment [48]. Therefore, compared to deep learning, reinforcement learning focuses more on learning problem-solving strategies.

Taking hierarchical reinforcement learning (HRL) as an example, HRL can solve different problems in the same system through hierarchical tasks or strategies. Try to consider the problem of attention dilution in the NAIS model. User

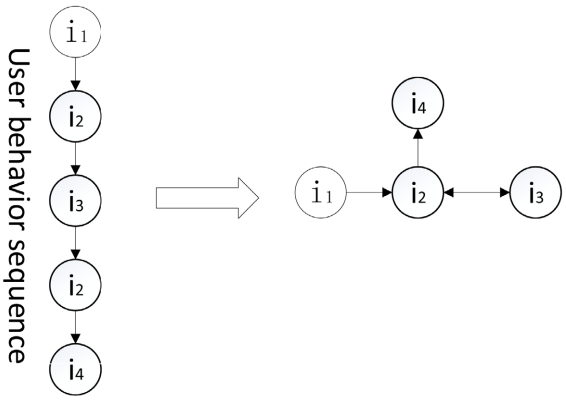

Figure 5. Sequence graph.

historical noise items will dilute the attention weight of the user's truly relevant items. Irrelevant items that are not related to majors will even get higher prediction scores than truly related items, leading the system to recommend items that do not meet user preferences. Zhang *et al.* [49] proposed that by removing noise items from user history items, a better recommendation effect can be achieved. Therefore, they built a hierarchical network structure and used NAIS model or NASR [40] model as the basic model to train the denoised data. By building a reinforcement learning framework on the upper layer of the basic model, building a user profile modifier, trying to eliminate irrelevant items to obtain higher rewards, thereby constructing a new user profile. Experimental results show that HRL has achieved a significant improvement compared to NAIS and other models. However, the model does not consider the different order of the interaction sequence, and ignores the user's dynamic interest at different times, which leads to a decrease in the effect of the model. In order to take the user's dynamic interest into account, DARL [50] designed a dynamic attention mechanism based on hierarchical reinforcement learning, which automatically captured users' dynamic preferences in each interaction and adaptively updated the attention weight of corresponding items in the session to improve the adaptability of the model. It can be seen that the HRL method decomposes the final goal into multiple levels of subtasks, processes the subtasks of each layer separately to learn hierarchical strategies, and then combines the strategies of each subtask to form an effective global strategy. In many complex recommendation tasks, the strategy learning directed by the final goal often leads to low efficiency, but the hierarchical method can effectively improve the recommendation efficiency.

In addition to hierarchical reinforcement learning, some frontier research directions of current reinforcement learning also include multi-task transfer reinforcement learning, multi-agent reinforcement learning, and reinforcement learning based on memory and reasoning, etc.

5. Commonly Used Datasets and Evaluation Metrics

5.1. Datasets

In the course recommender system, the data used is mainly the interaction record between the user and the course, that is, the record information of the user's course selection. The dataset is used to evaluate the effectiveness of the recommendation method, so a good large-scale dataset is very important for the research of the recommender system. However, there are not many public datasets currently used in the course recommender system. The most commonly used datasets include MOOCCourse and MOOCCube, both of which are collected and organized from XuetangX. Among them, the MOOCCourse data set was compiled by Zhang *et al.* [49], and it contains 82,535 users and 1302 courses related to course selection records, and each user has registered at least 3 courses. The historical average number of courses per user is 5.55. The MOOCCube dataset is collated from [51] which is an open data repository serving mas-

sive open online courses related research. The student behavior records include the student behavior information such as learning duration, times of learning and the interval of learning videos. The student behavior records include student behavior information such as learning duration and video viewing times. In the course recommender system, the information usually used is only the information of the user's course selection record, so we only pay attention to the information of students' course selection. MOOCCube contains 55,203 users and 706 courses, and each user has registered at least 4 courses. On average, each user enrolls in 6.42 courses. The detailed statistics of these data sets are shown in Table 1.

5.2. Evaluation Metrics

The evaluation of recommendation results is an important part of recommender system research. The performance of a recommendation algorithm in the evaluation metrics reflects its performance. The most commonly used evaluation metrics in current course recommender systems include: Precision (P), Recall (R), Hit Rate (HR) and Normalized Discounted Cumulative Gain (NDCG).

Precision: The proportion of the number of correct items recommended to the total number of recommendations. The definition of precision is as follows:

$$\text{Precision} = \frac{\sum_{u=1}^{U}\left|R(u)\cap T(u)\right|}{\sum_{u=1}^{U}\left|R(u)\right|} \tag{4}$$

where $R(u)$ is a list of recommendations made to the user based on the user's behavior on the training set, and $T(u)$ is the list of user behaviors on the test set.

Recall: The ratio of the number of items that meet the user's requirements in the recommended results to all the numbers that meet the requirements. The definition of recall is as follows:

$$\text{Recall} = \frac{\sum_{u=1}^{U}\left|R(u)\cap T(u)\right|}{\sum_{u=1}^{U}\left|T(u)\right|} \tag{5}$$

Hit Rate: It is a recall-based metric that represents the percentage of items in the test set that were successfully recommended to users. The definition of hit rate is as follows:

$$\text{HR}@\text{K} = \frac{\sum_{u=1}^{U}\text{Hits}_u@\text{K}}{\left|GT\right|} \tag{6}$$

Table 1. Course recommender system dataset.

Dataset	Users	Courses	Interactions	Average interactions
MOOC Course	82,535	1302	458,453	5.55
MOOC Cube	55,203	706	354,541	6.42

where GT represents the ground-truth set in the test set, and $\text{Hits}_u@\text{K}$ represents the number of top-k recommended items of user u belonging to the test set, and $|\cdot|$ represents the size of the set.

Normalized Discounted Cumulative Gain: It is a precision-based metric used to explain the predicted position of different users' recommendation lists. The definition of NDCG@K is as follows:

$$\begin{aligned} \text{NDCG}@\text{K} &= \frac{1}{U}\sum_{u=1}^{U}\frac{\text{DCG}_u@\text{K}}{\text{IDCG}_u@\text{K}} \\ \text{DCG}_u@\text{K} &= \sum_{i=1}^{K}\frac{2^{rel_u^i}-1}{\log_2(i+1)} \end{aligned} \tag{7}$$

where DCG is used to measure the relevance of the recommendation result, and rel_u^i represents the relevance of the item at position i in the recommendation result to the user u. The more relevant items are arranged in the front of the recommendation list, the better the recommendation effect, and the greater the value of DCG. When the recommendation lists are all sorted by relevance, the DCG of this sequence is IDCG, which represents the ideal discount cumulative income of the best recommendation list obtained by the user.

6. Conclusions and Future Directions

Both artificial intelligence and recommender systems have been research hotspots in the past decade. A large number of relevant new technologies and new models appear every year, and the foregoing various recommended models have empirically proved their superior recommendation quality. We hope that this survey will give researchers a comprehensive overview of the most cutting-edge course recommendation model, and help readers to fully understand the key aspects of the field. At present, recommendation systems have been widely used in various applications such as e-commerce and social media, and have become an important part of online services in these application fields. However, the current course recommendation program is far from satisfactory, and there are still many opportunities in this field. We discuss some possible directions, from the data used, to the combination and improvement of various models, which deserve more research efforts.

Dataset: Although both MOOCCourse and MOOCCube provide useful student selection records, their data sources are all from XuetangX, which is not enough to represent the current online learning environment. In addition, the amount of data provided for students' selection of courses is still relatively small, which easily leads to problems such as insufficient generalization ability of the trained model. Therefore, the current research on the course recommender system still needs more high-quality and large-scale datasets to effectively evaluate the recommendation effects of various models.

Data category used: The current course recommender system mainly uses the user's historical course selection record data, without considering other available side information; For example, the student's own gender, age, major and other

information, or information that reflects the quality of the course, such as course introduction, category, instructor, etc. If these side information can be taken into consideration, technologies such as knowledge graphs can be introduced to further enhance the effect of course recommendation.

Hybrid model: Many current researches use a single type of recommendation algorithm without considering the characteristics of other algorithms. If the advantages of various algorithms can be comprehensively considered and different recommendation algorithms can be combined, the effect of online course recommendation may be further improved.

Acknowledgements

The author is grateful to Jinan University for encouraging them to do this research.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Shah, D. (2019) Year of MOOC-Based Degrees: A Review of MOOC Stats and Trends in 2018. Class Central.

[2] Salehi, M. and Kamalabadi, I.N. (2013) Hybrid Recommendation Approach for Learning Material Based on Sequential Pattern of The Accessed Material and The Learner's Preference Tree. *Knowledge-Based Systems*, **48**, 57-69. https://doi.org/10.1016/j.knosys.2013.04.012

[3] Woods, D.D., Patterson, E.S. and Roth, E.M. (2002) Can We Ever Escape from Data Overload? A Cognitive Systems Diagnosis. *Cognition, Technology & Work*, **4**, 22-36. https://doi.org/10.1007/s101110200002

[4] Ricci, F., Rokach, L., Shapira, B. and Kantor, P.B. (2011) Introduction to Recommender Systems Handbook. In: Ricci, F., Rokach, L., Shapira, B. and Kantor, P.B., Eds., *Recommender Systems Handbook*, Springer, Berlin, 1-35. http://dx.doi.org/10.1007/978-0-387-85820-3_1

[5] Lu, J., Wu, D., Mao, M., Wang, W. and Zhang, G. (2015) Recommender System Application Developments: A Survey. *Decision Support Systems*, **74**, 12-32. https://doi.org/10.1016/j.dss.2015.03.008

[6] Kumar, A. and Sharma, A. (2013) Alleviating Sparsity and Scalability Issues in Collaborative Filtering Based Recommender Systems. *Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications* (*FICTA*), Bhubaneswar, 14-16 November 2013, 103-112. https://doi.org/10.1007/978-3-642-35314-7_13

[7] Singh, P., Ahuja, S., Jaitly, V. and Jain, S. (2020) A Framework to Alleviate Common Problems from Recommender System: A Case Study for Technical Course Recommendation. *Journal of Discrete Mathematical Sciences and Cryptography*, **23**, 451-460. https://doi.org/10.1080/09720529.2020.1728899

[8] Drachsler, H., Hummel, H.G. and Koper, R. (2008) Personal Recommender Systems for Learners in Lifelong Learning Networks: The Requirements, Techniques and

Model. *International Journal of Learning Technology*, **3**, 404-423. https://doi.org/10.1504/IJLT.2008.019376

[9] Farzan, R. and Brusilovsky, P. (2011) Encouraging User Participation in a Course Recommender System: An Impact on User Behavior. *Computers in Human Behavior*, **27**, 276-284. https://doi.org/10.1016/j.chb.2010.08.005

[10] Isinkaye, F.O., Folajimi, Y.O. and Ojokoh, B.A. (2015) Recommendation Systems: Principles, Methods and Evaluation. *Egyptian Informatics Journal*, **16**, 261-273. https://doi.org/10.1016/j.eij.2015.06.005

[11] Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. (2001) Item-Based Collaborative Filtering Recommendation Algorithms. *Proceedings of the* 10*th International Conference on World Wide Web*, Hong Kong, 1-5 May 2001, 285-295. https://doi.org/10.1145/371920.372071

[12] Koren, Y., Bell, R. and Volinsky, C. (2009) Matrix Factorization Techniques for Recommender Systems. *Computer*, **42**, 30-37. https://doi.org/10.1109/MC.2009.263

[13] Rendle, S., Freudenthaler, C., Gantner, Z. and Schmidt-Thieme, L.B. (2014) Bayesian Personalized Ranking from Implicit Feedback. In: *Proceedings of Uncertainty in Artificial Intelligence*, AUAI Press, Arlington, Virginia, 452-461.

[14] He, K., Zhang, X., Ren, S. and Sun, J. (2016) Deep Residual Learning for Image Recognition. 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, 27-30 June 2016, 770-778. https://doi.org/10.1109/CVPR.2016.90

[15] Zhang, H., Yang, Y., Luan, H., Yang, S. and Chua, T.S. (2014) Start from Scratch: Towards Automatically Identifying, Modeling, And Naming Visual Attributes. In *Proceedings of the* 22*nd ACM International Conference on Multimedia*, Orlando, 3-7 November 2014, 187-196. https://doi.org/10.1145/2647868.2654915

[16] Collobert, R. and Weston, J. (2008) A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. *Proceedings of the* 25*th International Conference on Machine Learning*, Helsinki, 5-9 July 2008, 160-167. https://doi.org/10.1145/1390156.1390177

[17] Tay, Y., Anh Tuan, L. and Hui, S.C. (2018) Latent Relational Metric Learning via Memory-Based Attention for Collaborative Ranking. *Proceedings of the* 2018 *World Wide Web Conference*, Lyon, 23-27 April 2018, 729-739. https://doi.org/10.1145/3178876.3186154

[18] He, X., Liao, L., Zhang, H., Nie, L., Hu, X. and Chua, T.S. (2017) Neural Collaborative Filtering. *Proceedings of the* 26*th International Conference on World Wide Web*, Perth, 3-7 April 2017, 173-182. https://doi.org/10.1145/3038912.3052569

[19] Sedhain, S., Menon, A.K., Sanner, S. and Xie, L. (2015) Autorec: Autoencoders Meet Collaborative Filtering. *Proceedings of the* 24*th International Conference on World Wide Web*, Florence, 18-22 May 2015, 111-112. https://doi.org/10.1145/2740908.2742726

[20] Wu, Y., DuBois, C., Zheng, A.X. and Ester, M. (2016) Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, San Francisco, 22-25 February 2016, 153-162. https://doi.org/10.1145/2835776.2835837

[21] Kabbur, S., Ning, X. and Karypis, G. (2013) FISM: Factored Item Similarity Models for Top-N Recommender Systems. *Proceedings of the* 19*th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Chicago, 11-14 August 2013, 659-667. https://doi.org/10.1145/2487575.2487589

[22] Koren, Y. (2008) Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. *Proceedings of the* 14*th ACM SIGKDD International Confe-*

rence on Knowledge Discovery and Data Mining, Las Vegas, 24-27 August 2008, 426-434. https://doi.org/10.1145/1401890.1401944

[23] He, X., He, Z., Song, J., Liu, Z., Jiang, Y.G. and Chua, T.S. (2018) NAIS: Neural Attentive Item Similarity Model for Recommendation. *IEEE Transactions on Knowledge and Data Engineering*, **30**, 2354-2366. https://doi.org/10.1109/TKDE.2018.2831682

[24] Chen, J., Zhang, H., He, X., Nie, L., Liu, W. and Chua, T.S. (2017) Attentive Collaborative Filtering: Multimedia Recommendation with Item-And Component-Level Attention. *Proceedings of the* 40*th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Tokyo, 7-11 August 2017, 335-344. https://doi.org/10.1145/3077136.3080797

[25] Xue, F., He, X., Wang, X., Xu, J., Liu, K. and Hong, R. (2019) Deep Item-Based Collaborative Filtering for Top-N Recommendation. *ACM Transactions on Information Systems* (*TOIS*), **37**, 1-25. https://doi.org/10.1145/3314578

[26] Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., *et al.* (2018) Deep Interest Network for Click-Through Rate Prediction. *Proceedings of the* 24*th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, London, 19-23 August 2018, 1059-1068. https://doi.org/10.1145/3219819.3219823

[27] Kipf, T.N. and Welling, M. (2016) Semi-Supervised Classification with Graph Convolutional Networks. arXiv:1609.02907.

[28] Li, Q., Han, Z. and Wu, X.M. (2018) Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. *Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, 2-7 February 2018, 3538-3545.

[29] Wang, X., He, X., Wang, M., Feng, F. and Chua, T.S. (2019) Neural Graph Collaborative Filtering. *Proceedings of the* 42*nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Paris, 21-25 July 2019, 165-174. https://doi.org/10.1145/3331184.3331267

[30] He, X., Deng, K., Wang, X., Li, Y., Zhang, Y. and Wang, M. (2020) Lightgcn: Simplifying and Powering Graph Convolution Network for Recommendation. *Proceedings of the* 43*rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Virtual Event, 25-30 July 2020, 639-648. https://doi.org/10.1145/3397271.3401063

[31] Fang, H., Zhang, D., Shu, Y. and Guo, G. (2020) Deep Learning for Sequential Recommendation: Algorithms, Influential Factors, and Evaluations. *ACM Transactions on Information Systems* (*TOIS*), **39**, 1-42. https://doi.org/10.1145/3426723

[32] He, R. and McAuley, J. (2016) Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. 2016 *IEEE* 16*th International Conference on Data Mining* (*ICDM*), Barcelona, 12-15 December 2016, 191-200. https://doi.org/10.1109/ICDM.2016.0030

[33] Rendle, S., Freudenthaler, C. and Schmidt-Thieme, L. (2010) Factorizing Personalized Markov Chains for Next-Basket Recommendation. *Proceedings of the* 19*th International Conference on World Wide Web*, Raleigh, 26-30 April 2010, 811-820. https://doi.org/10.1145/1772690.1772773

[34] Hidasi, B., Karatzoglou, A., Baltrunas, L. and Tikk, D. (2015) Session-Based Recommendations with Recurrent Neural Networks. arXiv:1511.06939.

[35] Hidasi, B. and Karatzoglou, A. (2018) Recurrent Neural Networks with Top-K Gains for Session-Based Recommendations. *Proceedings of the* 27*th ACM International Conference on Information and Knowledge Management*, Torino, 22-26 October 2018, 843-852. https://doi.org/10.1145/3269206.3271761

[36] Wu, C.Y., Ahmed, A., Beutel, A., Smola, A.J. and Jing, H. (2017) Recurrent Recommender Networks. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, Cambridge, 6-10 February 2017, 495-503. https://doi.org/10.1145/3018661.3018689

[37] Quadrana, M., Karatzoglou, A., Hidasi, B. and Cremonesi, P. (2017) Personalizing Session-Based Recommendations with Hierarchical Recurrent Neural Networks. *Proceedings of the Eleventh ACM Conference on Recommender Systems*, Como, 27-31 August 2017, 130-137. https://doi.org/10.1145/3109859.3109896

[38] Tang, J. and Wang, K. (2018) Personalized Top-N Sequential Recommendation Via Convolutional Sequence Embedding. *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, Marina Del Rey, 5-9 February 2018, 565-573. https://doi.org/10.1145/3159652.3159656

[39] Yuan, F., Karatzoglou, A., Arapakis, I., Jose, J.M. and He, X. (2019) A Simple Convolutional Generative Network for Next Item Recommendation. *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, Melbourne, 11-15 February 2019, 582-590. https://doi.org/10.1145/3289600.3290975

[40] Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T. and Ma, J. (2017) Neural Attentive Session-Based Recommendation. *Proceedings of the* 2017 *ACM on Conference on Information and Knowledge Management*, Singapore, 6-10 November 2017, 1419-1428. https://doi.org/10.1145/3132847.3132926

[41] Liu, Q., Zeng, Y., Mokhosi, R. and Zhang, H. (2018) STAMP: Short-Term Attention/Memory Priority Model for Session-Based Recommendation. *Proceedings of the* 24*th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, London, 19-23 August 2018, 1831-1839. https://doi.org/10.1145/3219819.3219950

[42] Kang, W.C. and McAuley, J. (2018) Self-Attentive Sequential Recommendation. 2018 *IEEE International Conference on Data Mining* (*ICDM*), Singapore, 17-20 November 2018, 197-206. https://doi.org/10.1109/ICDM.2018.00035

[43] Zhang, S., Tay, Y., Yao, L., Sun, A. and An, J. (2019) Next Item Recommendation with Self-Attentive Metric Learning. *Thirty-Third AAAI Conference on Artificial Intelligence*, Honolulu, 27 January-1 February 2019, Vol. 9.

[44] Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W. and Jiang, P. (2019) BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. *Proceedings of the* 28*th ACM International Conference on Information and Knowledge Management*, Beijing, 3-7 November 2019, 1441-1450. https://doi.org/10.1145/3357384.3357895

[45] Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X. and Tan, T. (2019) Session-Based Recommendation with Graph Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, Honolulu, 27 January-1 February 2019, 346-353.

[46] Xu, C., Zhao, P., Liu, Y., Sheng, V.S., Xu, J., Zhuang, F., *et al.* (2019) Graph Contextualized Self-Attention Network for Session-Based Recommendation. *International Joint Conference on Artificial Intelligence*, Macao, 10-16 August 2019, 3940-3946.

[47] Ma, C., Ma, L., Zhang, Y., Sun, J., Liu, X. and Coates, M. (2020) Memory Augmented Graph Neural Networks for Sequential Recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence*, New York, 7-12 February 2020, 5045-5052.

[48] Sutton, R.S. and Barto, A.G. (1999) Reinforcement Learning: An Introduction. *Robotica*, **17**, 229-235. https://doi.org/10.1017/S0263574799211174

[49] Zhang, J., Hao, B., Chen, B., Li, C., Chen, H. and Sun, J. (2019) Hierarchical Reinforcement Learning for Course Recommendation in Moocs. *Proceedings of the AAAI Conference on Artificial Intelligence*, Honolulu, 27 January-1 February 2019, 435-442.

[50] Lin, Y., Feng, S., Lin, F., Zeng, W., Liu, Y. and Wu, P. (2021) Adaptive Course Recommendation in Moocs. *Knowledge-Based Systems*, **224**, Article ID: 107085. https://doi.org/10.1016/j.knosys.2021.107085

[51] Yu, J., Luo, G., Xiao, T., Zhong, Q., Wang, Y., Feng, W., *et al*. (2020) Mooccube: A Large-Scale Data Repository for NLP Applications in Moocs. *Proceedings of the* 58*th Annual Meeting of the Association for Computational Linguistics*, Online, 5-10 July 2020, 3135-3142.