Knowledge-Based Systems 210 (2020) 106481

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Exercise recommendation based on knowledge concept prediction

Zhengyang Wu [a,b,*], Ming Li [c], Yong Tang [a,b,**], Qingyu Liang [a]

[a] School of Computer Science, South China Normal University, Guangzhou, China
[b] Guangzhou Key Laboratory of Big Data and Intelligent Education, Guangzhou, China
[c] Department of Educational Technology, Zhejiang Normal University, Jinhua, China

ARTICLE INFO

Article history:
Received 16 April 2020
Received in revised form 14 September 2020
Accepted 18 September 2020
Available online 8 October 2020

Keywords:
Recommender system
Online learning
Recurrent Neural Networks
Deep Knowledge Tracing

ABSTRACT

Good recommendation for difficulty exercises can effectively help to point the students/users in the right direction, and potentially empower their learning interests. It is however challenging to select the exercises with reasonable difficulty for students as they have different learning status and the size of exercise bank is quite large. The classic collaborative filtering (CF) based recommendation methods rely heavily on the similarities among students or exercises, leading to recommend exercises with mismatched difficulty. This paper proposes a novel exercise recommendation method, which uses Recurrent Neural Networks (RNNs) to predict the coverage of knowledge concepts, and uses Deep Knowledge Tracing (DKT) to predict students' mastery level of knowledge concepts based on the student's exercise answer records. The predictive results are utilized to filter the exercises; therefore, a subset of exercise bank is generated. As such, a complete list of recommended exercises can be obtained by solving an optimization problem. Extensive experimental studies show that our proposed approach has advantages over some existing baseline methods, not only in terms of the evaluation of difficulty of recommended exercises, but also the diversity and novelty of the recommendation lists.

1. Introduction

Online exercise is an important step in e-learning. In order to consolidate the learning effect of students at a certain stage, corresponding exercises are often provided to them before the beginning of the stage or after the end of the stage. Therefore, making the exercises appropriately challenging is one of the main goals of the adaptive online learning system. It is a challenge to recommend exercises with suitable difficulty, variety of types, and the knowledge concepts contained therein meet the requirements of learning progress. In recent years, various recommendation algorithms have been used to solve the problem of exercise recommendation. Similar to the application of traditional recommender systems in the field of e-commerce, these methods treat the students in the learning system as users in the e-commerce system, the exercises as commodity items, and the answers of the students to the exercises as the user's rating on the items. Therefore, the exercise recommendation problem can be solved using the methods commonly employed in the e-commerce recommender system [1]. The core idea of these methods is to first find similar students (users) or exercises (items), then predict the result of answer of target students based on the similar students or similar exercises, and then recommend exercises based on the prediction results. It shows that the prediction of whether the target student answers the exercise correctly or not is the main basis for the exercise recommendation. However, the existing recommendation algorithms cannot directly recommend exercises with appropriate difficulty for the target student because manually assigning the difficulty label unavoidably leads to label bias.

Generally, an exercise contains many attributes, of which the knowledge concept coverage and difficulty of the exercise are the most important attributes. Obviously, the knowledge concept is the backbone embedded in various entities which appears during the learning process, and an exercise can be seen as an evaluation of a student's mastery of one or more knowledge concepts. In addition, the results of whether the student answer the exercise correctly or not will affect his/her learning engagement. Authors in [2] studied empirically this issue by conducting extensive experiments with varied difficulty of exercises that are used a e-learning system. Their results indicate that the short-term engagement is better in case of easier exercises (e.g., error rate is close to 10%) while the long-term engagement is better in case of more difficult exercises (e.g., error rate is close to 50%). Different student may have a different feeling of difficulty on the same exercise, although each exercise has a stable difficulty

* Corresponding author at: School of Computer Science, South China Normal University, Guangzhou, China.
** Corresponding author at: Guangzhou Key Laboratory of Big Data and Intelligent Education, Guangzhou, China.
E-mail addresses: wuzhengyang@m.scnu.edu.cn (Z. Wu), yongt@m.scnu.edu.cn (Y. Tang).

https://doi.org/10.1016/j.knosys.2020.106481
0950-7051/© 2020 Elsevier B.V. All rights reserved.

label in some exercise banks. The root cause is that the degree of mastery of the corresponding knowledge concept is distinct for each student. Moreover, for a student, the exercises he/she once found difficult to answer will become easier with the enhancement of his/her mastery of the corresponding knowledge concepts. On the other hand, learning is the process of mastering knowledge concepts on a regular basis, so the exercises should follow the rule, and cannot be separated from the personal learning progress [3]. Therefore, both the coverage and mastery of the knowledge concepts are needed in the process of building exercise recommendation system [4]. Also, the consideration of the serendipity is helpful for the exercise recommendation since the exercise diversity and novelty have a certain degree of influence on the student's learning enthusiasm [5].

In this paper, we propose a novel Exercise Recommendation method based on Knowledge Concept Prediction, termed KCP-ER. Basically, the whole framework of KCP-ER consists of two modules: the Knowledge Concept Prediction Layer (KCPL) that is used for obtaining the embedding of the knowledge concept coverage and the mastery of students, and the Exercise Set Filtering Layer (ESPL) that aims at filtering the exercises based on the corresponding prediction results produced by KCPL. For problem-solving in KCPL, two modules are included, that is, one LSTM-based module for the prediction of the knowledge concept coverage prediction (KCCP), and one Deep Knowledge Tracing [6] module for prediction of the knowledge concept mastery (KCMP). Two new types of loss function with consideration of the knowledge concept are used as the optimization objective for KCCP and KCMP, respectively. In the ESPL module, we first build an exercise bank filter to produce an exercise set including exercises with suitable difficulty and novelty, then develop an knowledge-concept-based simulated annealing algorithm to generate an exercise recommendation list, by which the diversity of the exercise recommendation can be guaranteed. Our experimental study demonstrates that the recommended list of exercises obtained by our approach has better diversity and novelty. In particular, the difficulty of the recommended exercised are closer to the desired one, as compared with several baselines. Overall, our contribution can be summarized as the following three-fold:

- A novel exercise recommendation approach based on knowledge concept prediction is developed. Both the LSTM-based module for the prediction of the knowledge concept coverage, and the Deep Knowledge Tracing module for prediction of the knowledge concept mastery are proposed for model design. Experimental results show that our proposed model outperforms significantly than some existing baselines.
- A knowledge concept prediction layer is designed to obtain the unified embedding of the knowledge concept coverage and the mastery of students. This can ensure that the recommended exercises have the desired difficulty even in the case that difficulty labels are missing or have certain level of biases.
- A knowledge-concept-based simulated annealing algorithm with special consideration of evaluating Jaccard distance of the exercises is developed to generate an exercise recommendation list, which potentially increase the diversity of the recommended exercises.

The rest of the paper is organized as follows. We briefly review the related work in Section 2. In Section 3, we introduce some preliminaries and important notations. In Section 4, we describe the framework and implementation of KCP-ER. Extensive experimental studies are conducted in Section 5. Finally, we give the conclusion in Section 6.

2. Related work

In this section, we first review the related research about the exercise recommendation. Then, we present a basic explanation on the meaning of the difficulty of exercises and revisit some existing methods for exercise difficulty prediction.

2.1. Exercise recommendation

In the domain of educational data mining, research on recommendation of personalize learning resources has received careful attention. Technically, the traditional recommendation methods such as the Content-Based Filtering (CBF) [7,8], the Collaborative Filtering (CF) [9,10] and the Hybrid Filtering (HF) [11,12] recommendation methods are widely used in learning resource recommendation. A content-based exercise recommendation method is proposed in [13], which is based on the similarity of attributes between the exercise and the learning target. A method for recommending a balanced exercise for a student's interactive exercise answering system is proposed in [14], which can recommend new exercises to qualified students in areas of interest and expertise. However, due to the differences in behavioral motivation, the recommendation scheme used in the learning system is significantly different from the typical e-commerce recommendations [15].

There are also some works considering the gap between student learning level and learning objectives in the process of building recommendation system. Authors in [16] propose a model for recommending exercises based on the student historical behavior as well as the personal learning objectives and expected learning paths. In [17], a recommendation system is built by equipping with a cognitive diagnostic based model that is developed to analyze the gap between the students' mastery of knowledge concepts and their learning goal. Besides, some works focus on the relationship within the students and learning resources. A hybrid filtering recommendation method is proposed in [18], which combines CF and CBF using multiple criteria related both to student and course information to recommend the most suitable courses to the students. In [19], a knowledge-based e-learning recommendation method is proposed, which uses the semantic ontology to represent students and learning resources. Authors in [20] use association rules, by which the hidden relationships between learners' course activities are extracted, to develop a course recommendation model.

Recently, the deep learning-based recommender systems are investigated with the application on e-learning recommendation [16,21]. These models represent a large amount of data related to students and learning resources by training deep neural networks. Feature learning is automatically performed on source heterogeneous data, and different data is mapped into the same hidden space to obtain a deep unified feature representation of students and learning resources. In [22], a dual-regularized matrix factorization with deep neural networks model is proposed, to exploit textual information for building recommender systems. In addition, the graph-based model is an important progress in the research of recommender system [23]. Some studies pay more attention to the graph-based recommendation method. In these methods, user behavior data is represented in the form of graphs. For instance, in [24], a knowledge graph-based recommendation approach is proposed for learning path recommendation.

2.2. Exercise difficulty and prediction

One of the core goals of adaptive e-learning systems is making exercises suitably challenging, and it is generally believed that a good exercise should be neither easy nor too difficult. Easy exercises lead to better short term engagement, whereas difficult

Table 1
Some important notations.

Notation	Description
C	The course in the learning system
EB	The bank of course exercise
K	The list of knowledge concept of the course
$k \in K$	A knowledge concept in the knowledge concept list
$e(K) \in EB$	The exercise in the exercise bank
$H = \langle e(K), a \rangle$	The two-tuple of exercise answer pair

exercises are better for the long term engagement [2]. It can be inferred that the difficulty of exercise is the crucial attribute for the students able to adhere to the learning system. The exercises in the learning system may have preset difficulty labels. Since this label is set manually, it must be subjective. But in practice, different students may experience different difficulties with the same exercise. This is due to students' mastery of knowledge concept and their proficiency in using knowledge. What is more, not all exercises are labeled with difficulty. In this case, we can assess the difficulty of the exercise by predicting the probability that the student will answer the exercise correctly. On the other hand, in the scenario which recommend the course to a class or a group of students, we must consider the contribution of each student as a member of the group and their conflicts with each other [25]. Therefore, the prediction of each student's learning performance is necessary for learning recommendation.

There are many studies on student performance predictions, some based on the student's emotional analysis [26], some based on the student's behavioral analysis [27], and some based on the student's exercises answered records analysis [28]. Where an available method to predict performance based on exercise history is named Knowledge Tracing. The original Knowledge Tracing model is based on the Bayesian network, so it is also called BKT [29]. BKT model divides the learning process into unlearned and learned two stages, and uses the Hidden Markov model (HMM) [30] to represent the probability distributions along the discrete timeline. In the literature [31], authors propose a model to adjust the two learning phase states of BKT to three learning phase states to improve the accuracy of prediction. In the literature [6], authors propose a novel Knowledge Tracing framework by considering RNN model for problem-solving, which is named Deep Knowledge Tracing (DKT). Basically, DKT model uses the exercise answer sequence data as the training data and the equipped RNN can robustly predict whether or not a student will solve a particular exercise correctly given the accuracy of historic solutions. In the literature [32], authors propose a method for predicting exercise difficulty using Convolutional Neural Networks (CNN) and RNN, which is based on trial text and answer records. Technically, these studies demonstrate that RNN-based models are generally effective in predicting learning performance.

3. Preliminary

In this section, we first introduce the way to represent an exercise by using knowledge concepts. Then, we illustrate the working process of how to estimate the difficulty of exercises, and how to abstain the recommended exercises by the prediction of the knowledge concepts. Some important notations are listed in Table 1, and we present more detailed explanation of their role in the context.

Given a course C with knowledge concept list K, and the corresponding exercise bank EB. The problem is to select appropriate $e(K)$ from EB to recommend to students who take the course C. In general, in an exercise bank, some exercises contain only one knowledge concept, and some contain two or more different knowledge concepts. Then, we can use the distribution of knowledge concept to represent an exercise, i.e.

$$e(K) = [e(k_1), e(k_2), \dots, e(k_n)],$$

where n is the length of K. If i-knowledge concept k_i is covered by $e(K)$, then $e(k_i) = 1$, otherwise $e(k_i) = 0$.

3.1. Coverage of knowledge concepts

An exercise is a summary of the knowledge concept of the course, which usually contains one or more knowledge concepts. Generally, reasonable exercises should include the knowledge concepts that student is currently learning. Therefore, the first task is to predict the knowledge concepts that should be included in the exercises appearing at next time according to the exercises completed by students from the beginning of learning to the present.

Given the student's record of answering exercises from time 0 to t, the problem is to predict the likelihood that a knowledge concept will appear at time $t+1$. We denote the record of answering exercises from time 0 to t by H_t. The appearance probability of ith knowledge concept k_i is denoted by $P(k_i^c)$. We denote the coverage of knowledge concepts in exercises at next time by:

$$P(K^c) = [P(k_1^c), P(k_2^c), \dots, P(k_m^c)],$$

where, $P(K^c)$ is a vector of knowledge concept coverage, and its length is the same as the length of K.

3.2. Exercise difficulty estimation

The probability that a student answers an exercise correctly can often be used to indicate the difficult level of the exercise for him/her. That is, when the student is more likely to answer the exercise correctly, he/she feels that the exercise is less difficult.

Given the records of a student's exercise answer from time 0 to t, the problem is to predict the probability of correctly answer exercise by he/she at time $t+1$. Whether a student answers an exercise correctly depends on the probability that he/she answers the knowledge concept in the exercise correctly. We denote the probability that he/she correctly answers all knowledge concepts at next time by

$$P(K^m) = [P(k_1^m), P(k_2^m), \dots, P(k_n^m)].$$

Since an exercise is a combination of a series of knowledge concepts, we can use the following equation to get the probability of correctly answering an exercise.

$$R_{e(K)} = \prod_{i=1}^{n} (P(k_i^m) | e(k_i) == 1), \tag{1}$$

where $R_{e(K)}$ represents the probability of correctly answering the exercise $e(K)$. We denote the difficulty of $e(K)$ by $D_{e(K)}$, then $D_{e(K)} = 1 - R_{e(K)}$.

We use *Deep Knowledge Tracing* [6] model to get $P(K^m)$, which adopted the trained LSTM networks. The process is illustrated in Fig. 1. The input x_t is the encoded form of the knowledge concept covered by an exercise answered by a student at time t and whether the answer is correct or not; the output y_t is a vector where the length equals to the length of K, and each component of it represents the probability of correctly answering the corresponding knowledge concept. After inputting the student's exercises record, the output of the trained LSTM network is his/her $P(K^m)$ for all knowledge concepts in the course.

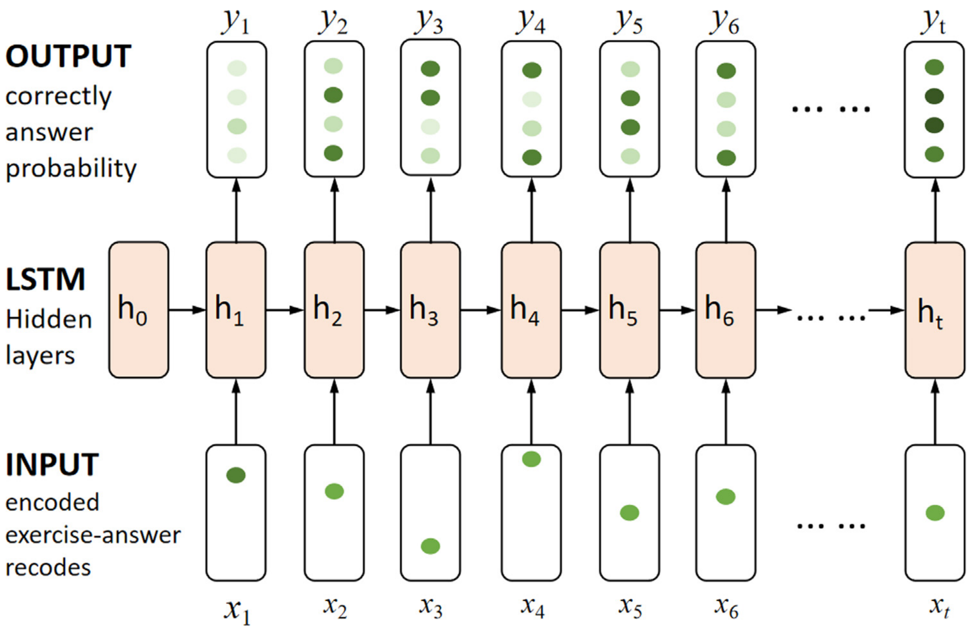
OUTPUT
correctly answer probability
LSTM
Hidden layers
INPUT
encoded exercise-answer recodes
y_1 y_2 y_3 y_4 y_5 y_6 y_t
h_0 h_1 h_2 h_3 h_4 h_5 h_6 h_t
x_1 x_2 x_3 x_4 x_5 x_6 x_t

Fig. 1. Process of predicting the probability that a student correctly answers all knowledge concepts.

3.3. Filtering of exercises

Given the coverage of knowledge concepts in exercises $P(K^c)$ and the difficulty of each exercise $D_{e(K)}$. The problem is to filter out a set of exercises from EB and generate a list of recommended exercises. According to the previous analysis, the knowledge concepts included in the exercises in the recommendation list should follow the requirements for knowledge concept coverage in learning progress. At the same time, the difficulty of these exercises should be as close as possible to the desired difficulty. Then, the filter conditions for the exercises in EB can be formalized as:

$$\arg\max_{e(K)\in EB}(cossim(e(K), P(K^c))) \wedge \arg\min_{e(K)\in EB}(distance(\delta, D_{e(K)})), \quad (2)$$

where we use the cosine similarity $cossim$ to compare $e(k)$ and $P(K^c)$. δ is the desired difficulty of recommended exercise, and $distance(\delta, D_{e(K)})$ represents the distance between the desired difficulty and the prediction difficulty.

4. Proposed method for exercise recommendation

In this section, we propose a novel method for exercise recommendation by using the prediction of knowledge concepts. For simplicity in the following presentation, we use a short name "KCP-ER" for our proposed method. The framework of KCP-ER is shown in Fig. 2. As shown in the figure, KCP-ER is a double-layer architecture model that takes student's exercise answer records as input and the recommended exercises list (REL) as output. The first layer is named *Knowledge Concept Prediction Layer* (KCPL), and the second layer is named *Exercise Set Filtering Layer* (ESFL). The KCPL is used to predict the coverage and mastery of knowledge concepts according to the student's exercise answer record, as the basis for selecting appropriate recommended exercises in the next step. The ESFL is used to generate a list of recommended exercises based on the knowledge concept prediction results output by KCPL. The details of KCPL and ESFL will be illustrated in Sections 4.1 and 4.2.

4.1. Knowledge concept prediction layer

This module is used to predict the coverage and mastery of knowledge concepts as the basis for selecting appropriate recommended exercises in the next step. As Fig. 2 shows, two sub-modules inside KCPL are used to achieve this task. One is *KC Coverage Prediction* (KCCP), which is used to predict the coverage of the knowledge concept next time. The results of the prediction are denoted by the rectangle marked $P(K^c)$. The other is *KC Mastery Prediction* (KCMP), which is used to generate the probability of correctly answering knowledge concepts. The sub-module KCMP is implemented adopts the DKT model. The results of the prediction are denoted by the rectangle marked $P(K^m)$. Both $P(K^c)$ and $P(K^m)$ will be represented as vectors.

4.1.1. Knowledge concepts coverage prediction

We predict the appearance probability of each knowledge concept at time $t+1$ by using the occurrence sequence of knowledge concepts from time 0 to t. The knowledge concept coverage prediction is a sequence-based prediction problem. Therefore, we adopt a LSTM networks to obtain a sequence prediction model. Fig. 3 shows the process of predicting knowledge concept coverage by LSTM networks. Where the round rectangle marked A^t represents the state of the model at time t. The input rectangle K^t represents the knowledge concept appeared at time t, and the output rectangle K^{t+1} represents the probability of knowledge concept appeared at time $t+1$.

The training data of the LSTM is a knowledge concept appearance sequence extracted from students' exercise answer records. We denote this sequence by K^T, where T represents a period of time during which a student doing exercise, and $t \in T$ represents each step in T. Before inputting the network, K^t should be transferred into the one-hot encoding, which is denoted by $\phi(K^t)$. The output of the model is a vector whose size is equal to the number of knowledge concepts, which is denoted by z^t. During the training period T, z^t implies the prediction of the next knowledge concept, and the next knowledge concept is exactly $\phi(K^{t+1})$, which is also the input to the model at time $t+1$. Therefore, we use binary cross-entropy loss to construct a loss

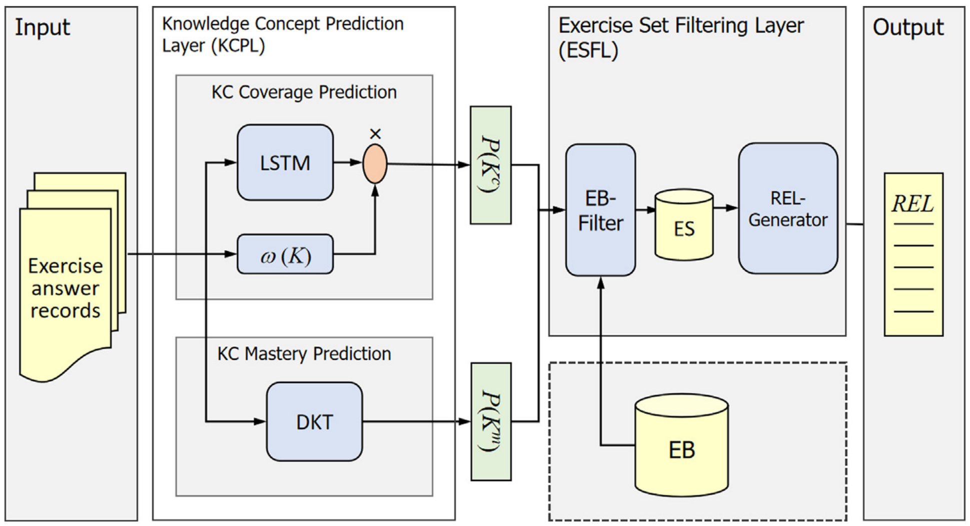

Fig. 2. Framework of KCP-ER.

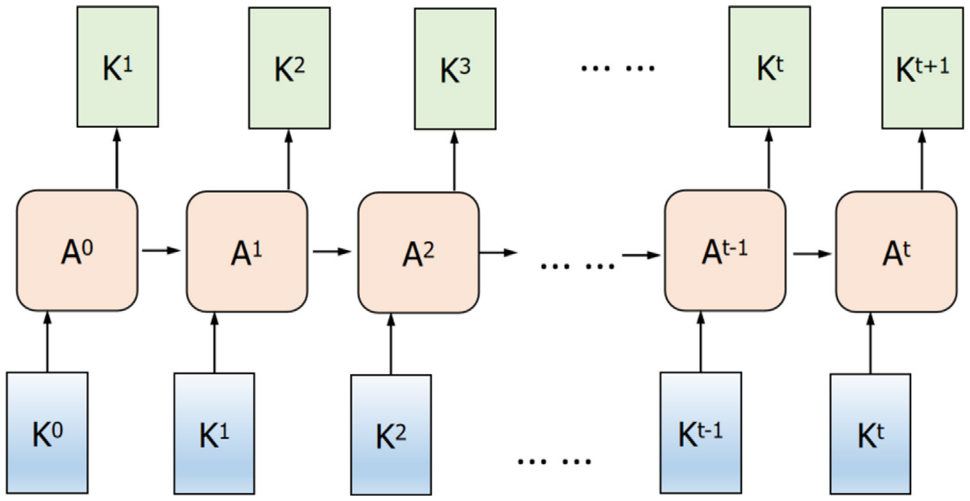

Fig. 3. The process of predicting knowledge concept coverage by LSTM model.

function to train this model. The optimized loss function for a single student can be expressed as:

$$\mathcal{L}_{s(c)} = \sum_{t=0}^{T} l_b(z^t \cdot \phi(K^{t+1}), 1), \tag{3}$$

where the symbol $\cdot$ indicates the dot-product, l_b represents binary cross-entropy loss. Then, we will express the optimized cost function of the KCCP as:

$$\mathcal{L}_{KCCP} = \frac{1}{\sum_{i=1}^{n} T_i} \sum_{i=1}^{n} \sum_{t=0}^{T_i} l_b(z_i^t \cdot \phi(K^{t+1}), 1), \tag{4}$$

The output of the LSTM model is a vector whose length is equal to the number of knowledge concepts in the course, which is denoted by

$$c(K) = [c(k_1), c(k_2), \dots, c(k_m)],$$

where each element of it is the appearance probability of the corresponding knowledge concept at time $t+1$. At the same time, we hope that those knowledge concepts that have been answered correctly appear as little as possible in future exercises. Therefore, we add a weight vector to the output of the LSTM networks, whose length is equal to the number of knowledge concepts. We denote this weight vector by

$$\omega(K) = [\omega(k_1), \omega(k_2), \dots, \omega(k_m)],$$

and the element of $\omega(K)$ can be calculated as

$$\omega(k_i) = \begin{cases} 1 - \dfrac{r_i}{c_i}, & c_i > 0 \\ 1, & c_i = 0, \end{cases} \tag{5}$$

where r_i is the number of correct answers to k_i, and c_i is the occurrence number of k_i. Then, the $P(K^c)$ can be obtained using the output of *LSTM* networks and the weight vector in the sense that,

$$P(k_i^c) = c(k_i)\omega(k_i), \tag{6}$$

where, $P(k_i^c)$ is the ith element of $P(K^c)$.

4.1.2. Knowledge concept mastery prediction

The probability of correctly answering the knowledge concepts reflects the student's mastery level of the corresponding knowledge concepts. In this sub-module, we use the DKT model to predict students' knowledge concept mastery. According to the optimization method of DKT, the training objective is the negative log-likelihood of the observed sequence of student answers under the model. We denote the one-hot encoding of the knowledge concept answered at time $t+1$ by $\phi(K^{t+1})$, and denote the corresponding answering by a^{t+1}. The output of DKT at time t is denoted by y^t. The loss function is constructed by the binary-cross entropy, which for a single student can be expressed as:

$$\mathcal{L}_{s(m)} = \sum_{t=0}^{T} l_b(y^t \cdot \phi(K^{t+1}), a^{t+1}), \tag{7}$$

where the symbol $\cdot$ indicates the dot-product, l_b represents binary cross-entropy loss. Then, the optimization cost function of the KCMP can be expressed as:

$$\mathcal{L}_{KCMP} = \frac{1}{\sum_{i=1}^{n} T_i} \sum_{i=1}^{n} \sum_{t=0}^{T_i} l_b((y_i^t) \cdot \phi(K^{t+1}), a^{t+1}), \tag{8}$$

where n is the number of students, and T_i indicates the time period for the ith student to answer previous exercises.

4.2. Exercise set filtering layer

The major target of our method is to recommend exercises with suitable difficulty, on the other hand, it also need to consider the serendipity of recommendation. The reason is that knowledge concepts that have been correctly answered by the student in the past should rarely appear in the recommendation list. Therefore, we hope to improve the novelty of knowledge concepts of recommended exercises on the basis of ensuring the appropriate difficulty. In order to solve this problem, we adopt a pre-filtering method, that is, a condition that combines suitable difficulty and novelty is first generated, which is used to filter exercises. On the basis of ensuring appropriate difficulty and novelty, we adopt the post-filtering method, which re-rank the recommendation lists according to their diversity. The *Exercise Set Filtering Layer*(ESFL) suppose to filter exercise from *EB*. The process of filtering includes two steps. The first step is to filter the exercises according to the probability of knowledge concepts, and produce an exercise set which includes the exercises with suitable difficulty and novelty. The second step is to generate a list of recommended exercises (REL) from the exercise set produced by EB-Filter to improve the diversity of recommendation.

4.2.1. Exercise bank filter

The *KCPL* module generates two types prediction, where $P(K^c)$ represents the knowledge concepts coverage at next time, and $P(K^m)$ represents the probability of the knowledge concepts answered correctly. Both $P(K^c)$ and $P(K^m)$ are sent to an *Exercise Bank Filter* (EB-Filter), in order to produce a set of exercises that meet the knowledge concept coverage and difficulty requirements. Fig. 4 shows the internal structure of EB-Filter.

During the process of filtering, the exercises in *EB* are randomly sampled and input one by one to EB-Filter. The rounded rectangle marked *Similarity* represents a process of comparing the similarity between the exercise $P(K^c)$ and $e(K)_i$. We denote this similarity by $Sim(P(K_n), e(K)_i)$. The square marked δ indicates the desired difficulty of the exercise. The rounded rectangle marked *Distance* represents a process of calculating the distance between the prediction difficulty of exercise and δ. Here, we get the probability of correctly answering $e(K)_i$ by Equ. (1), and further get the prediction difficulty $D_{e(K)_i}$. The distance between $D_{e(K)_i}$ and δ is denoted by $Dis(\delta, D_{e(K)_i})$. The rectangle marked $\Omega_{e(K)_i}$ represents the weight of the exercise e_i, which can be got by calculating the Euclidean Norm of $Sim(P(K_n), e(K)_i)$ and $Dis(\delta, D_{e(K)_i})$, i.e,

$$\Omega_{e(K)_i} = \sqrt{Sim(P(K^c), e(K)_i)^2 + Dis(\delta, D_{e(K)_i})^2}. \tag{9}$$

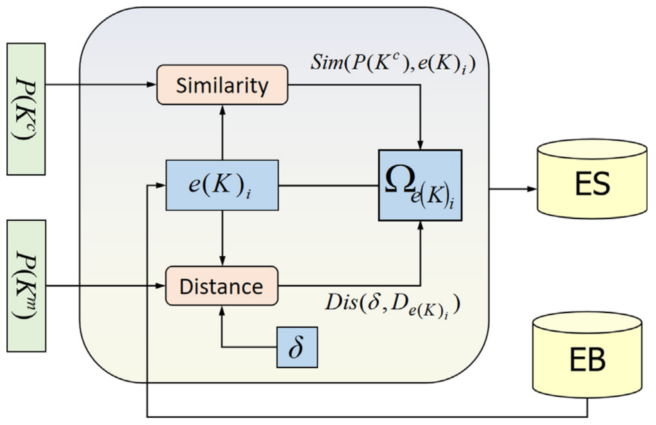

Fig. 4. Framework of EB-Filter.

After obtaining the weight of each exercise $\Omega_{e(K)_i}$, this module sorts exercises by increasing the weight and selects N exercises with the smallest weight to form an exercise set, which is denoted by *ES*. *ES* can be expressed as:

$$ES = \{e(K)_i \in EB | \Omega_{e(K)_i} \le \Omega_{e(K)_N}\}, \tag{10}$$

where $\Omega_{e(K)_N}$ represents the Nth weight value sorted by increasing weight.

Algorithm 1 summarizes the process of generating the *ES* via EB-filter. The loop is the process of calculating the weight of each exercise. Line 7 is the process of sorting the exercises by weight. Line 8 represents the process of forming ES by select N exercises according to their weights.

Algorithm 1 EB-filter Algorithm

```
Input: $EB$, $P(K_n)$, $P(K_c)$, $\delta$, $N$
Output: $ES$
1: for $i \le size(EB)$ do
2:     $e_i \longleftarrow Sample(EB, 1)$ {Select an exercise from $EB$}
3:     $sim \longleftarrow Cosine - Similarity(P(K_n), e_i)$ {Get the similarity}
4:     $dis \longleftarrow \delta - D_{e_i}$ {Get the distance of difficulty}
5:     $\Omega_{e_i} \longleftarrow \sqrt{sim^2 + dis^2}$ {Get the weight of each exercise}
6: end for
7: $sort(e, \Omega_e)$ {Sort exercises by weight}
8: $ES \longleftarrow Select(e, \Omega_e \le \Omega_{e_N})$ {Form $ES$}
```

Algorithm 1 traverses the EB firstly, and then sorts all exercises once, the number of executions $T(n) = n + n = 2n$, therefore, its time complexity is $O(n)$.

4.2.2. Recommended exercises list generator

We use *KCPL* and EB-Filter to ensure that the exercises in *ES* have the desired difficulty and reasonable knowledge concept coverage. In order to improve the diversity of recommendation, we set a sub-module named *Recommended Exercises List Generator* (REL-generator). This module is used to generate a subset of ES with high diversity.

We consider *ES* as a high-dimensional space, and each exercise represents a point in that space. Then, our task is translated to select the M points with the largest distance in this space that is a combinatorial optimization problem. Therefore, we use simulated annealing algorithm [33] to handle it. The framework of the REL-generator is shown in Fig. 5. In the upper left part of the rounded rectangle, the rectangle marked L_0 represents the initialized version of REL. It is the one with the largest distance in REL composed of randomly selected exercises in *ES*. The content

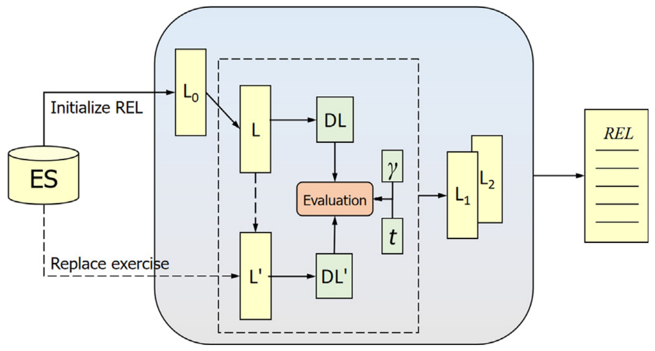

Fig. 5. Framework of the REL-generator.

in the dotted box represents the process of updating the REL by using the simulated annealing algorithm. The rectangle marked L represents a version of REL that was originally copied from L_0. The rectangle marked L' represents an updated version of L. The squares marked DL and DL' represent the exercise distance matrices of L and L' respectively. DL and DL' are square matrices whose column and row sizes are equal to the number of exercises in REL. Two small squares marked γ and $\mathcal{T}$ represent two parameters of the simulated annealing algorithm, where $\mathcal{T}$ represents temperature and γ is a random number in range 0 to 1. The rounded rectangle marked *Evaluation* represents the process of evaluating the DL and DL'. If the $DL \le DL'$ the L' will be accepted as a new version REL. Otherwise, L' will be accepted with probability p. p can be calculated by the following equation:

$$p = \exp\left(\frac{-(mean(DL') - mean(DL))}{\kappa_B \mathcal{T}}\right), \tag{11}$$

where $mean(DL)$ represents the mean of the distance matrix of list L, κ_B is the *Boltzmann constant*. When $\gamma \ge p$, RQL L' will be accepted. We compare the distance matrix mean of each accepted REL and keep the version with the largest distance matrix mean as the output of REL-generator.

Algorithm 2 summarizes the process of generating REL via REL-generator from *ES*. The inputting parameter $\mathcal{T}$ represents temperature, parameter κ_B is the *Boltzmann constant*, and parameter c represents the reduction factor. First, the algorithm generates M RELs by randomly selecting exercises from *ES*. Then, the one with the largest distance is selected as the first version REL. The *FOR* loop is the process of *simulated annealing*. First, we replace some of the exercises in the original L to get L'. If the exercises distance of L' large than L, we accept L' as the new version L_{new}. Otherwise, we calculate the acceptance probability p and compare it with a randomly generated decimal γ to decide whether to accept L'. The *WHILE* loop is used to save different versions of REL after the processing of *simulated annealing*. After satisfying the termination criterion, we chose the L with the largest distance as the REL of output.

In Algorithm 2, it is natural to obtain the computational complexity of the iterations between Step 4 and Step 17, i.e, $O(n)$. With consideration of the iterative process indicating in Step 3, which is used for the termination criterion, the overall computational complexity of Algorithm 2 is $O(m \times n)$.

4.3. Explanation of the model

In KCP-ER, we use the three features, i.e., knowledge concept mastery, diversity and novelty, to evaluate the list of recommended exercises. Among them, the knowledge concept mastery is generated by the RNN in the KCMP module, the novelty is adjusted by $\omega(K)$ in the KCCP module, and the diversity is adjusted by the REL-generator in ESFL. Since the ES obtained by the first filtering depends entirely on the output of the KCMP and KCCP modules, the exercises in the recommended list are highly likely to depend on the knowledge concept mastery and novelty. In addition, the diversity of the recommended list depends on the differences between the exercises contained in the ES, and the optimal combination can only be selected in a limited search space, so the degree of influence on the resulting recommended exercises list is relatively low.

Technically, our approach has good potential for dealing with item-cold-start issues. In practice, the difficulty labels of exercises are most easily missed and biased due to manual labeling. When we add a new exercise into the EB, the exercise's knowledge concepts and type are clear, but there is a high possibility that the difficulty labels are missing or contain biases. The proposed approach can effectively cope with the item-cold-start problem because it utilizes the knowledge concepts of the new exercise to predict the possibility of the event that the students answer it correctly, with no need for its difficulty label or considering whether it has been answered by the other students. For the user-cold-start problem, we will introduce its solution in Section 5.3.4.

Algorithm 2 REL-generator Algorithm

```
Input: $ES$, $M$, $\mathcal{T}$, $\kappa_B$ ,$c$
Output: $REL$
 1: Randomly generate $M$ RELs from $ES$
 2: $L \longleftarrow$ The one with largest distance in $M$ RELs
 3: while termination criterion is not satisfied do
 4:   for $i \le$ iteration number do
 5:     $L' \longleftarrow Replace(L, e)$ {Replace exercise to get a new version}
 6:     if $mean(D_{L'}) \ge mean(D_L)$ then
 7:       $L_{new} \longleftarrow L'$ {Save the $L'$ as the new version}
 8:     else
 9:       $p \longleftarrow \exp \frac{-(mean(DL')-mean(DL))}{\kappa_B t}$
10:       $\gamma \longleftarrow random(0, 1)$
11:     end if
12:     if $\gamma \ge p$ then
13:       $L_{new} \longleftarrow L'$ {Save the $L'$ as the new version}
14:     else
15:       $L_{new} \longleftarrow L$ {Save the $L$ as the new version}
16:     end if
17:   end for
18:   $L \longleftarrow L_{new}$
19:   $\mathcal{T} \longleftarrow c \times \mathcal{T}$
20: end while
21: $REL \longleftarrow \arg\max_L(mean(D_L))$ {Form $ES$}
```

5. Experiments

In this section, we conduct extensive experimental studies to verify the effectiveness and advantage of our approach. We compare the recommendation performance produced by KCP-ER with several baselines on the different datasets.

5.1. Datasets

Our experiments are performed on three classic exercise-answer datasets. An overview of the datasets is shown in Table 2.

- **ASSISTments**
  This dataset [34] is an electronic tutor that teaches and evaluates students in grade-school math. The dataset we used was gathered in the 2009–2010 school year. There is no separate exercise bank for this dataset, but fortunately

Table 2
Overview of Data sets.

Data set	KCs	Students	Exercises	Records
ASSISTments 2009–2010	124	4,217	17,751	278,607
Algebra 2005–2006	437	574	1,085	809,694
OLIES 2011	87	335	300	45,002

each exercise answer record shows some attributes of the corresponding exercise. These attributes include the identification of the exercise, the knowledge concepts associated with the exercise, the template of the exercise, and the type of answer to the exercise. Then, we separate exercises from this dataset and form an exercise bank containing 17,751 exercises, each one of which has four attributes.

- **KDDCup Algebra**
This dataset [35] comes from the KDD Cup 2010 educational data mining challenge, which is an algebra exercise sequence dataset collected in 2005–2006. The dataset has 809,694 records, 437 Knowledge concepts, 574 students and 1085 exercises.
- **OLI Engineering Statics**
This dataset [36] was collected from a college-level engineering statics course in 2011. It has 45,002 records, 87 knowledge concepts, 335 students and 300 exercises. This dataset is available from the PSLC DataShop website.

These datasets have on exercise difficulty label data, so we simulate the difficulty label for each exercise based on the correct rate of each knowledge concept in the exercise answer records. The difficulty label $difficulty_{e(K)}$ of the exercise $e(K)$ can be expressed as:

$$difficulty_{e(K)} = \frac{\sum_{i=1}^{n}(correct_i|e(k_i) == 1)}{\sum_{i=1}^{n} e(k_i)}, \tag{12}$$

where n is the number of knowledge concepts of the course, $e(k_i)$ represents whether the ith knowledge concept appears or not in the exercise $e(K)$ (as mentioned in Section 3), $correct_i$ is the correct rate of ith knowledge concept.

5.2. Experimental setup

In this subsection, we first clarifying the details of implementation to set up the framework of KCP-ER. Then, we introduce the comparison baselines and evaluation metrics of the experiments.

5.2.1. Implementation details

LSTM model training In order to train the LSTM network, we first one-hot encoding the data. The input of the LSTM network in KCCP is the one-hot encoding of the knowledge concept sequence. Each element of the sequence is a vector whose length is the number of knowledge concepts. If the vector represents the ith knowledge concept, its ith bit is 1 and all other bits are 0. The input of the DKT model is the one-hot encoding of the tuple of knowledge concept and answer, i.e., $\langle K, a\rangle$, which is a vector whose length is the double of the number of knowledge concepts. If ith knowledge concept is answered correctly, then the vector's ith and $(i + m)$th bit are 1, all other bits are 0. If ith knowledge concept is answered incorrectly, then its ith bit is 1 and all other bits are 0. LSTM in KCCP and DKT have the same architecture. Each hidden layer contains 200 fully connected hidden nodes. In order to speed up the training process, we used a mini-batch of stochastic gradient descent to minimize the loss function, and the implemented batch size is 64. We trained the model at a learning rate of 0.01, and also applied dropout to avoid overfitting. After training by iteration 100 epochs for each model, the AUC of the DKT model on ASSISTments2009–2010 is 0.799, on Algebra 2005–2006 is 0.737, and on OLIES2011 is 0.701. The LSTM network model in KCCP obtained the AUC on ASSISTments2009–2010 is 0.901, on Algebra 2005–2006 is 0.899, and on OLIES2011 is 0.871. The accuracy of the DKT model is different on different datasets, which in practice measures the difference in the accuracy of the prediction of knowledge concept mastery. As the accuracy of prediction improves, our approach can better ensure that the recommended exercises meet the desired difficulty. And the accuracy of the KCCP module can ensure that the exercises recommended by our approach contain reasonable knowledge concepts.

Exercises filtering We use 70% of the dataset to train the two LSTM networks in the KCPL module, using 10% as the validation dataset, and the remaining 20% as the test dataset. In other words, in each dataset, 20% of the student data is used to test the performance of the model. Specifically, in the ASSISTments2009–2010 dataset, we random select 300 students from the test students set, in the Algebra 2005–2006 dataset, the number of test students is 115, and in the OLIES 2011 dataset, the number of test students is 71. We input each student's exercise answer sequence to the KCCP and KCMP sub-modules, get the probability vectors $P(K^c)$ and $P(K^m)$. The EB-Filter calculates a weight for each exercise in EB based on $P(K^c)$ and $P(K^m)$. Next, the EB-Filter collects exercises according to the weights from small to large, and generate the ES. We set EB as a sub-exercise bank with 150 exercises, i.e., $N = 150$. The accuracy of the EB-Filter can ensure that the exercises recommended meet the requirements of desired difficulty and knowledge concepts coverage at the same time.

Largest difference exercises founding We select a set of exercises from ES through the REL-generator sub-module and form a list of recommended exercises. In this stage, we set the number of exercises in REL of 10, i.e., $M = 10$. Then, a set of 10 exercises is used to construct a 10×10 symmetric matrix, and the average value of the matrix is used as the optimization factor of the Algorithm 2. We set the initial temperature parameter $\mathcal{T}$ to 100, and set the reduction factor c to 0.095. The number of iterations of the inner loop is 100, and the termination criterion of the outer loop is that the temperature reaches 0.

5.2.2. Baselines

The user-based collaborative filter and the item-based collaborative filter are two classic algorithms of the recommender system [37]. Therefore, we use the student-based collaborative filter method [10] (SB-CF) and the exercise-based collaborative filter method [38] (EB-CF) as baselines. The other baseline is Content-based Filtering (CBF), which is a method based on the similarity of exercise and object of recommended exercise. Besides these classic methods, we also use two state-of-the-art methods as baselines, one is a hybrid recommendation model based on deep collaborative filtering [21] (HB-DeepCF), and the other is a knowledge-graph embedding based collaborative filtering [39] (KGEB-CF).

- The SB-CF method is based on student similarity recommendations. First, the method construct a student–student similarity matrix according to the exercise answer records. Then, it identifies Top10 students who have answered similar exercises to the target students. Next, the method finds out the one whose difficulty label value is closest to the desired difficulty from the exercises that each similar student answered, and add it to the recommendation list.
- The EB-CF method first sets the distance of difficulty to the weight of each exercise answered by the student. Then, the similarity vectors of all the exercises and the exercises that have been done are extracted from the exercise–exercise

similarity matrix, and the length of each vector is the number of exercises answered. Then calculate the weighted sum of the vector elements for each exercise, and recommend Top10 after sorting from large to small.

- The CBF method first calculates the distance between the difficulty label of each exercise and the desired difficulty for the weight of all exercises. Then, the method calculates the similarity of all the exercises with the exercises answered by the target student at time t. Next, the method calculates the weighted sum of the vector elements for all exercises, and recommend Top10 after sorting from large to small.
- The HB-DeepCF method is a hybrid recommendation method, which integrating auto-encoder with classical recommendation model. This method uses the auto-encoder to generate the latent vectors representing the exercises and students. The parameters of the recommender component and the auto-encoder component are adjusted at the same time. During the training process, the recommender component is used to provide guidance for the auto-encoder to learn more semantic features.
- The KGEB-CF method takes students and exercises as entities, and the results of students answering exercises as relationships. Through knowledge-graph embedding, a low-dimensional vector is obtained for each entity and relationship learning, the structure and semantic information of the original graph are maintained in the vector. Combined with the collaborative filtering recommendation method, the structure and semantic information of the graph is included in the recommendation process.

5.2.3. Evaluation metrics

The difficulty of exercises will affect the student engagement in learning, so the difficulty determines the quality of the recommended list. The novelty of the recommended list is to ensure that the exercises are effectively advanced with the knowledge concept, and the diversity is to increase students' interest in doing exercises. Our approach is to ensure that the recommended exercises have suitable difficulty and contain reasonable knowledge concepts, as much as possible to enrich the exercise types, so as to promote the enthusiasm of students to do exercises. The ground-truth of our approach is the result (correct/incorrect) of the student's answer to the exercise, and the recommended target is that the correct rate of the student's answer to the exercise is close to the set value (In the experiment, the value is 0.7). In practical applications, the actual difficulty of the recommended exercises for the students can be evaluated by the correct answer rate, which is calculated from the correct proportion of the students who answered all the recommended exercises after setting the desired difficulty.

The prediction accuracy is a traditional metric [40] for evaluating recommender systems. The purpose of this study is to recommend the desired difficulty exercises for students. Therefore, we first use the difficulty of recommended exercises to measure the *accuracy* indicator of methods. On the other hand, in the process of recommendation, the repeated appearance of the same knowledge concepts may confuse students, and the monotonous exercise types may also make students bored. This problem can be solved by increasing the serendipity of recommendation. Therefore, we also evaluate the *Serendipity* metrics of methods, namely *novelty* and *diversity*. However, increasing the serendipity of recommendation sometimes leads to inadequate results [41]. Therefore, we will show together with the results of the *accuracy* and *serendipity* indicators, and further analyze our approach.

- **Accuracy**
Note that one of our goals is to recommend exercises with desired difficulty δ. To test the accuracy of the recommendation, we calculate the relative distance between the difficulty of each exercise in the recommendation list and set the desired difficulty as $|\delta - D_{e(K)_i}|$. The value of this distance is between 0–1, and the smaller the value, the closer the difficulty of the exercise is to the desired difficulty, so we use $1 - |\delta - D_{e(K)_i}|$ to represent the accuracy of each recommended exercise. Then, the *Accuracy* of the entire recommendation list can be expressed as:

$$Accuracy(L_U) = \frac{\sum_i^U (D_{q(K)_i})}{|U|}, \tag{13}$$

where $D_{q(K)_i} = 1 - |\delta - D_{e(K)_i}|$, L_U is the REL which has U exercises. Furthermore, the results in [2] indicate that the short-term engagement is better in case of easier exercises (e.g., error rate is close to 10%) while the long-term engagement is better in case of more difficult exercises (e.g., error rate is close to 50%). In view of this, we choose the middle value in the experiment (i.e., error rate is 30%), that is, the desired difficulty is set to 0.7.

- **Novelty**
Novelty [42] metric reflects the serendipity of recommendations, and a novel item is a relevant or irrelevant item that a target user has never seen or heard before. In this study, we consider novel exercises to be ones that include knowledge concepts which not be answered or correctly answered by the target student. We denote a set of knowledge concepts covered by exercise $e(K)$ by $\mathcal{K}(e(K))$, which can be expressed as:

$$\mathcal{K}(e(K)) = \{k_i | e(k_i) == 1\}, \tag{14}$$

where $e(k_i) == 1$ represents the ith knowledge concept is covered by $e(k)$. Then, the set of knowledge concepts covered by correctly answered exercises in the exercise record can be expressed as:

$$\mathcal{K}(e(K)^r) = \bigcup_{e(k_i)^r \in H} \{k_i | e(k_i)^r == 1\}, \tag{15}$$

where $e(K)^r$ represents the exercise correctly answered. Before gaining novelty, we first define the distance of the exercises and express it as:

$$dist(e(K), e(K)^r) = 1 - Jaccsim(\mathcal{K}(e(K)), \mathcal{K}(e(K)^r)), \tag{16}$$

where $Jaccsim(\mathcal{K}(e(K)), \mathcal{K}(e(K)^r))$ represents the *Jaccard Similarity* [43] between $\mathcal{K}(e(K))$ and $\mathcal{K}(e(K)^r)$. The larger the $dist(e(K), e(K)^r)$ in the list of exercises, the novelty of the recommended exercises higher. Then the Novelty of the entire list can be expressed as:

$$Novelty(L_U) = \frac{1}{U} \sum_{u=1}^{U} dist(e(K)_u). \tag{17}$$

- **Diversity**
The Diversity metric represents the difference of the recommended exercises, which can be expressed by the average similarity of all exercises on the recommended exercise list [44]. Here we use the *cosine similarity* between two exercises to represent the different of them, which can be expressed as:

$$different(e(K)_i, e(K)_j) = 1 - \frac{e(K)_i \cdot e(K)_j}{||e(K)_i|| \; ||e(K)_j||}, \tag{18}$$

where $e(K)_i$ and $e(K)_j$ represent the different exercises in recommendation list. The larger $different(e(K)_i, e(K)_j)$ in the

Table 3
Comparisons of indicator *accuracy*.

	assistments0910		algebra0506		olies2011	
	Mean	Std.D	Mean	Std.D	Mean	Std.D
CBF	0.848	0.098	0.543	0.135	0.626	0.096
HB-DeepCF	0.826	0.128	0.697	0.210	0.859	0.103
KGEB-CF	0.887	0.097	0.637	0.115	0.871	0.064
EB-CF	0.483	0.200	0.470	0.131	0.801	0.078
KCP-ER	**0.899**	**0.079**	**0.866**	**0.052**	**0.894**	0.057
SB-CF	0.725	0.169	0.563	0.143	0.841	**0.052**

Table 4
Comparisons of indicator *diversity*.

	assistments0910		algebra0506		olies2011	
	Mean	Std.D	Mean	Std.D	Mean	Std.D
CBF	0.801	0.100	0.602	0.218	0.328	0.059
HB-DeepCF	0.766	0.150	0.624	0.189	0.769	0.183
KGEB-CF	0.535	0.182	0.680	0.214	0.885	0.085
EB-CF	0.289	0.203	0.332	0.162	0.421	0.165
KCP-ER	**0.772**	**0.080**	**0.751**	**0.086**	**0.948**	**0.029**
SB-CF	0.572	0.204	0.634	0.194	0.654	0.071

Table 5
Comparisons of indicator *novelty*.

	assistments0910		algebra0506		olies2011	
	Mean	Std.D	Mean	Std.D	Mean	Std.D
CBF	0.900	0.072	0.656	0.142	0.731	0.078
HB-DeepCF	0.917	0.087	0.741	0.082	0.874	0.066
KGEB-CF	0.918	0.074	0.685	0.108	0.871	0.059
EB-CF	0.952	0.064	0.696	0.109	0.841	0.078
KCP-ER	**0.963**	**0.025**	**0.839**	0.068	**0.883**	**0.044**
SB-CF	0.953	0.057	0.715	**0.062**	0.847	0.057

list, the more diverse the recommended exercises are. Then the Diversity of the entire list can be expressed as:

$$Diversity(L_U) = \frac{2\sum_{q(K)_i,q(K)_j\in L_u,i\neq j} different(q(K)_i, q(K)_j)}{|U|(|U|-1)}. \tag{19}$$

5.3. Results and discussion

This subsection illustrates the experimental results on the three datasets.

5.3.1. Comparison of individual indicators

We first show the results of accuracy indicator and serendipity indicator respectively. The serendipity indicators include novelty and diversity.

Tables 3–5 show the comparison of three indicators *accuracy*, *diversity* and *novelty* of the recommended exercise lists generated by using six methods on three datasets. The Mean marked column shows the average of the corresponding indicators of the recommendation lists. The closer the value is to 1, the better the performance of the method. The column marked Std.D shows the standard deviation of the indicators corresponding to the recommendation lists. The closer the value is to 0, the better the performance. As shown in these tables, KCP-ER has advantages over the other five baseline methods.

Fig. 6 shows the quality of generated lists of recommended exercise in terms of *accuracy* on the three datasets. Compared with baselines, the list of recommended exercises generated using KCP-ER is more advantageous in terms of accuracy. In other words, any exercise recommended using KCP-ER is much more likely to have the desirable difficulty than using baselines. This has been observed in all datasets.

Fig. 7 shows the quality of generated lists of recommended exercises in terms of *diversity* on the three datasets. It is clear that the list of recommended exercise generated using KCP-ER in most cases have a higher value. In dataset *algebra0506*, the KGEB-CF has more advantages than KCP-ER, but the curves of KCP-ER is smoother than that of KGEB-CF, as shown in Table 4. That said, the indicator *diversity* of KCP-ER is more stable.

Fig. 8 shows the quality of generated lists of recommended exercise in terms of *novelty* on the three datasets. The results show that most lists generated using KCP-ER have a higher value. That is to say, the novelty of the exercises recommended by KCP-ER is higher than the exercises recommended by the baselines.

Moreover, from Figs. 6–8, we can find that the experimental result curves of the KCP-ER method are flat compared to baselines. That is, the effectiveness of our method is more stable than the baseline.

5.3.2. Comparison of accuracy and serendipity

The serendipity improvement of recommendation sometimes leads to insufficient results [41]. This subsection shows the accuracy indicator together with the two serendipity indicators.

The quality of the recommendation lists in terms of *accuracy* and *novelty* on the three datasets are shown in Fig. 9. In the quadrant formed with accuracy as the ordinate axis and novelty as the abscissa axis, the upper right corner represents the better performance area in which *accuracy* value and *novelty* value both are closer to 1. Compared with baselines, the recommended lists generated using KCP-ER are more closely around the desirable area. Similar to Fig. 10, our approach can ensure that the recommended exercises have the desired difficulty, while maintaining the good novelty of the recommendation list.

Fig. 10 shows the quality of the recommendation lists in terms of *accuracy* and *diversity*. The quadrant shown in Fig. 10 is composed of the accuracy as the vertical axis and the diversity as the horizontal axis, and the upper right corner also represents a better performance area in which *accuracy* value and *diversity* value both are closer to 1. Fig. 10 is shown the recommended lists generated by KCP-ER are more closely around the desirable area.

5.3.3. Discussion

The above experimental results show that the recommendation list generated using KCP-ER is closer to the desired difficulty, and at the same time, it can maintain good novelty and diversity on the recommendation list. KCP-ER has advantages over the baselines in both accuracy and serendipity of recommendation. Obviously, serendipity has a great relationship with the size of the exercise bank. For example, in dataset ASSISTments2009–2010, because the number of exercises is large, the novelty of generating the recommendation lists are also relatively high. However, due to the small number of the types of exercise, the diversity of the generated recommendation lists is low. In dataset OLIES2011, although the number of exercises is small, the types are relatively rich, the diversity of the recommended list is relatively high. It can be seen that with our approach, the serendipity of recommendation will increase with the increasing of the number of exercises and the enrichment of exercise types, but the accuracy will not decrease.

The method of HB-DeepCF trains the recommender component and the auto-encoder component jointly. Compared with the traditional collaborative filtering method, the exercises and student latent vector representations generated by the auto-encoder component in this method have richer semantics than the ones used in the traditional collaborative filtering methods. Therefore, it can outperform the traditional collaborative filtering method in the performance comparison. The method of KGEB-CF takes the student (s) and the exercise (e) as entities, and the result

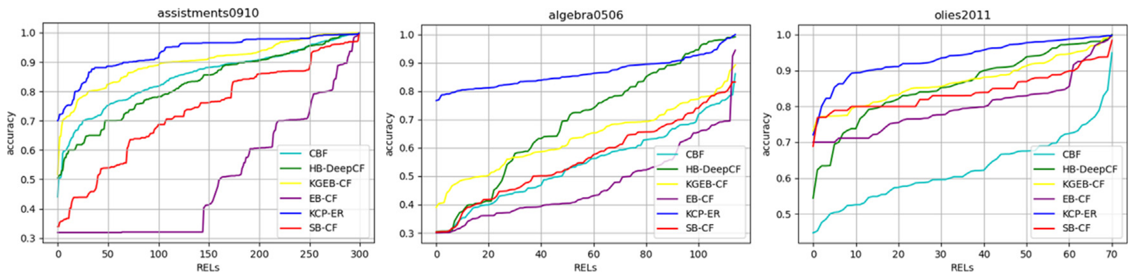

Fig. 6. Accuracy of recommended exercises.

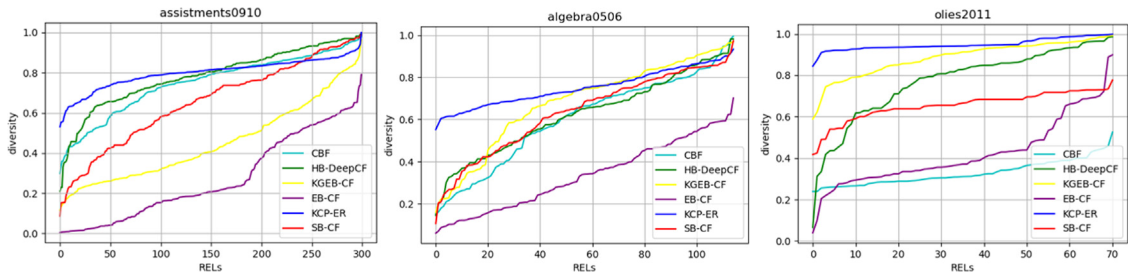

Fig. 7. Diversity of recommended exercises.

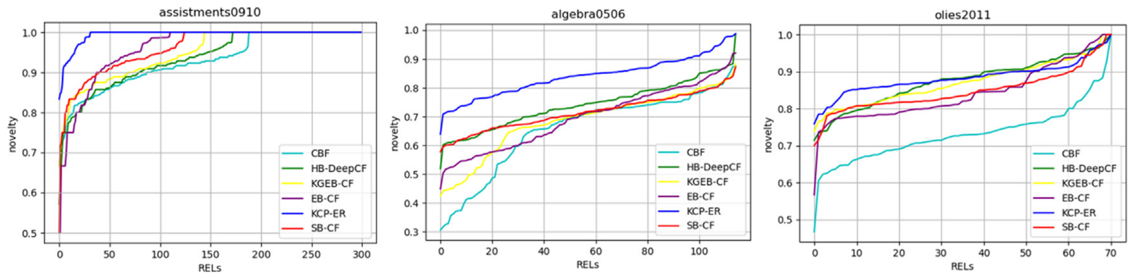

Fig. 8. Novelty of recommended exercises.

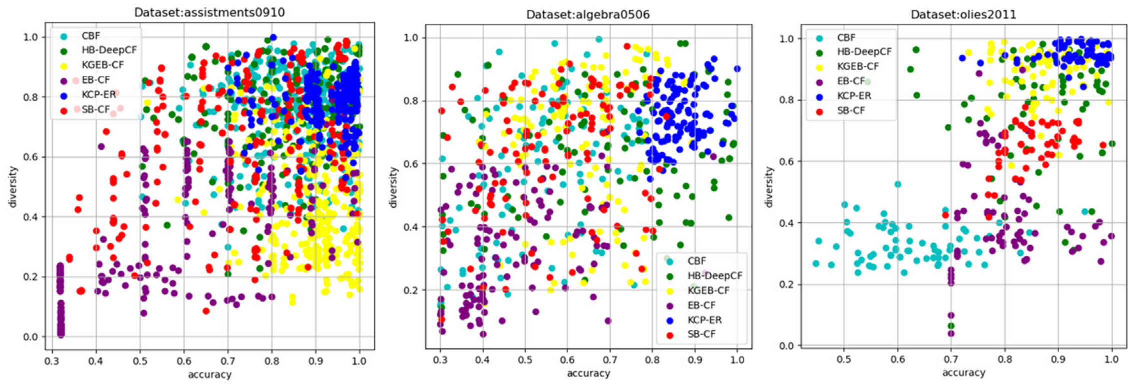

Fig. 9. Comparison of accuracy and novelty.

(r) of the exercises as the relationship. This method uses the *TransE* model to make ($s + e$) equal to r as much as possible by continuously adjusting the vectors s, e and r. The representation vectors generated by this method maintain the semantics of the correlation between the students, the exercises, and the results of the exercises; therefore, it can produce more semantics than the representation in the traditional collaborative filtering. However, the *DKT* module in our approach generates the vector representation of the exercise based on the prediction of the student's knowledge concept mastery, which contains richer semantic about the probability of correct answer for the exercise, so that the recommended exercises are closer to the desired difficulty. Furthermore, since the EB-filter is used to compare the similarity of latent vectors, and the REL-generator is used to ensure the diversity of the recommended list, the proposed method has advantages over baseline methods in terms of novelty and diversity.

5.3.4. Cold-start

The proposed approach can effectively cope with the item-cold-start problem because it utilizes the knowledge concepts of the new exercise to predict the possibility of the event that the students answer it correctly, with no need for its difficulty label or considering whether it has been answered by the other students.

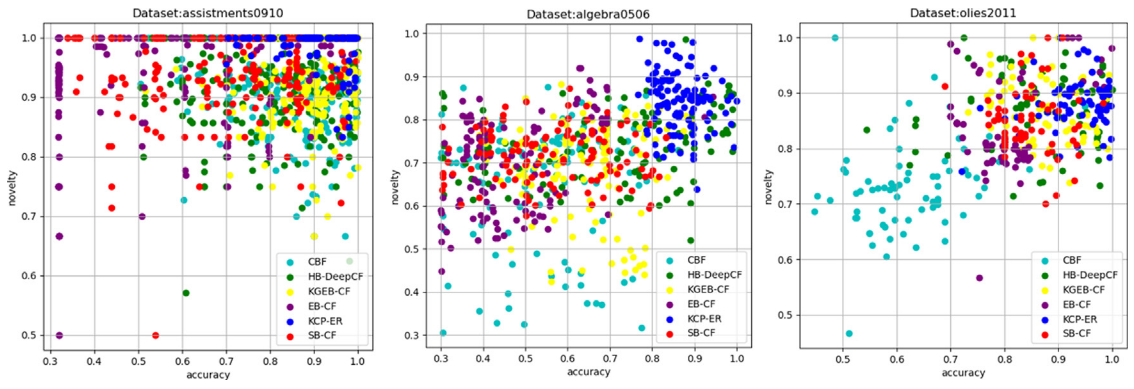

Fig. 10. Comparison of accuracy and diversity.

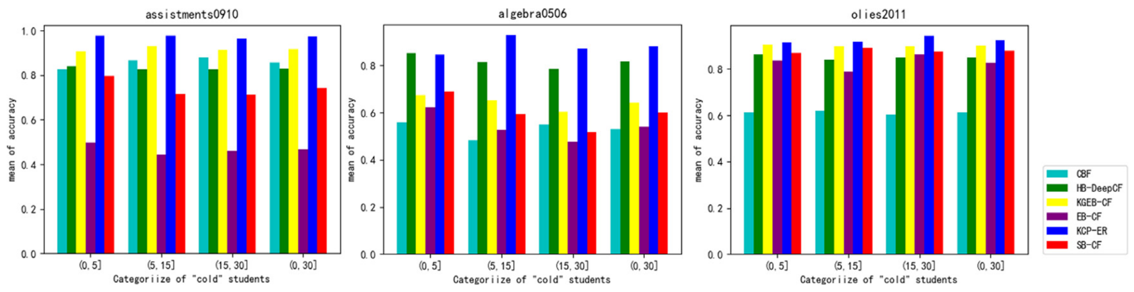

Fig. 11. Comparison of accuracy and diversity.

With respect to the user-cold-start problem, when a "cold" student enters the online exercise system, his/her answer records may be few, but the *KCPL* module still needs to generate two types of predictions based on his/her answer records. Therefore, the user-cold-start problem of the proposed approach is related to the cold-start problem of the knowledge concepts sequence prediction model and the knowledge tracking model. The *LSTM* networks used in the proposed approach has a memory function for short-term sequences. In addition, in literature [45], it is proposed that a model that integrates external attributes can be used to better alleviate the cold-start problem of knowledge tracking based on sequence prediction. According to this idea, we use the method of integrating answering records and some external attributes to match the most similar user for the "cold" student to alleviate the user-cold-start problem. The calculation equation of similarity between the "cold" students (is denoted by s_c) and "non-cold" students (is denoted by s_n) is:

$$\begin{aligned} sim_{s_c,s_n} &= \omega_c \cdot Sim_{DTW}(seq(\phi\langle K,a\rangle_{s_c}), seq(\phi\langle K,a\rangle_{s_n})) \\ &\quad + (1-\omega_c) \cdot Sim_{Euclidean}(v(ei)_{s_c}, v(ei)_{s_n}), \end{aligned} \tag{20}$$

where Sim_{DTW} is the *Dynamic Time Warping* [46] similarity between the exercises answered sequences, $\phi\langle K,a\rangle$ is the one-hot encoding of the tuple of knowledge concept and answer, i.e., $\langle K,a\rangle$. $v(ei)$ is a vector which represents the external attributes, and $Sim_{Euclidean}$ is the Euclidean similarity between the $v(ei)$ of the "cold" student and "non-cold" student. ω_c is a weight parameter.

We consider studying the performance w.r.t. different cold-start degrees, i.e., the exercise answering sparsity. To test it, we first categorize "cold" students from the test data into three groups according to the numbers of their exercise answering records, i.e., $(0,5]$, $(5,15]$ and $(15,30]$. In the data set ASSISTments0910, we selected 50 students for each "cold" group. In the data set Algebra0506, we selected 12, 27, and 38 students for each "cold" group. In the data set Olies2011, we selected 9, 16, and 26 students for each "cold" group. It is easy to see that the case for the first group is the most difficult, since students from this group have fewest exercise answering records. We present the performance comparison of different methods in Fig. 11, where the y-axis denotes the mean of accuracy. The proposed method performs the best among all the methods, and the improvement over SB-CF becomes more significant for users with fewer exercise answering records. The results show that the proposed KCP-ER method can utilize the embedding of knowledge concepts to retain more effective semantic information of the difficulty of exercises.

6. Conclusions

In this paper, we propose a novel approach (KCP-ER) for exercise recommendation by considering the prediction of knowledge concepts. To obtain a good embedding of exercise content, which is beneficial for making recommendation for the students, we first build a LSTM-based module for the prediction of the knowledge concept coverage prediction, and a Deep Knowledge Tracing module for the prediction of the knowledge concept mastery. Then, an exercise filtering module is developed to produce the final list of recommended exercises. We validate the advantages of KCP-ER on several real-world datasets used in educational data mining, in comparison with some existing methods in terms of accuracy of recommendation, as well as the diversity and novelty of the recommended exercises.

Recommendation of exercises with suitable difficulties, as an important function within the learning system, can help to keep or even motivate students' learning interests and promote continuous and effective learning activities. Although proposed approach takes into account the knowledge concepts of the exercises and the attributes of the type of exercises, it does not analyze the text information of the exercises or establish the semantic connection between the students' behavior and the exercises, which cannot fully ensure the students' interest in the process of doing exercise. In addition, our approach uses LSTM network to learn the occurrence of knowledge concepts based on the sequence of students' previous answers, and does not consider the internal connection between knowledge concepts. Regarding the cold-start problem, the RNN-based recommendation methods all need the training dataset. In future work, we can also use the cross-dimensional data from other learning management systems to address this issue [47]. In addition, we will aim to make the model more flexible and extensible. Also, the extension of our framework into an advanced version with further consideration of the multi-modal information of exercises, potentially enriching the knowledge concept distribution, is highlight expected.

CRediT authorship contribution statement

Zhengyang Wu: Conceptualization, Methodology, Writing. **Ming Li:** Data planning, Writing - original draft. **Yong Tang:** Visualization, Investigation. **Qingyu Liang:** Software, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (NSFC) under the Grant No. 61772211, U1811263, and 61802132. We used the ASSISTments 2009–2010 dataset accessed via Assistments website, the Algebra2005–2006 dataset accessed via KDD cup 2010 website, and the Oli engineering statics 2011 dataset accessed via DataShop. We would like to thank the anonymous reviewers for their constructive advice.

References

[1] JESUS Bobadilla, Francisco Serradilla, Antonio Hernando, et al., Collaborative filtering adapted to recommender systems of e-learning, Knowl.-Based Syst. 22 (4) (2009) 261–265.
[2] Jan Papoušek, Vít Stanislav, Radek Pelánek, Impact of question difficulty on engagement and learning, in: Proceedings of International Conference on Intelligent Tutoring Systems, Springer, 2016, pp. 267–272.
[3] Pin Lv, Xiaoxin Wang, Jia Xu, Junbin Wang, Utilizing knowledge graph and student testing behavior data for personalized exercise recommendation, in: Proceedings of ACM Turing Celebration Conference, ACM, 2018, pp. 53–59.
[4] Jiali Xia, Guangquan Li, Zhonghua Cao, Personalized exercise recommendation algorithm combining learning objective and assignment feedback, J. Intell. Fuzzy Systems 35 (3) (2018) 2965–2973.
[5] Denis Kotkov, Shuaiqiang Wang, Jari Veijalainen, A survey of serendipity in recommender systems, Knowl.-Based Syst. 111 (2016) 180–192.
[6] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J. Guibas, Jascha Sohl-Dickstein, Deep knowledge tracing, in: Proceedings of Advances in Neural Information Processing Systems, 2015, pp. 505–513.
[7] Andrew Walker, Mimi M. Recker, Kimberly Lawless, David Wiley, Collaborative information filtering: A review and an educational application, Int. J. Artif. Intell. Educ. 14 (1) (2004) 3–28.
[8] Michael J. Pazzani, Daniel Billsus, Content-based recommendation systems, in: The Adaptive Web, Springer, 2007, pp. 325–341.
[9] Greg Linden, Brent Smith, Jeremy York, Amazon.com recommendations: Item-to-item collaborative filtering, IEEE Internet Comput. 7 (1) (2003) 76–80.
[10] Gang Liu, Tianyong Hao, User-based question recommendation for question answering system, Int. J. Inf. Educ. Technol. 2 (3) (2012) 243–246.
[11] Aleksandra Klašnja-Milićević, Mirjana Ivanović, Alexandros Nanopoulos, Recommender systems in e-learning environments: a survey of the state-of-the-art and possible extensions, Artif. Intell. Rev. 44 (4) (2015) 571–604.
[12] Pei-Chann Chang, Cheng-Hui Lin, Meng-Hui Chen, A hybrid course recommendation system by integrating collaborative filtering and artificial immune systems, Algorithms 9 (3) (2016) 47–65.
[13] Khairil Imran Bin Ghauth, Nor Aniza Abdullah, Building an e-learning recommender system using vector space model and good learners average rating, in: Proceedings of the 9th IEEE International Conference on Advanced Learning Technologies, IEEE, 2009, pp. 194–196.
[14] Dawei Hu, Shenhua Gu, Shitong Wang, Liu Wenyin, Enhong Chen, Question recommendation for user-interactive question answering systems, in: Proceedings of the 2nd International Conference on Ubiquitous Information Management and Communication, ACM, 2008, pp. 39–44.
[15] Raciel Yera Toledo, Yailé Caballero Mota, Luis Martínez, A recommender system for programming online judges using fuzzy information modeling, Informatics 5 (2) (2018) 17.
[16] Tomohiro Saito, Yutaka Watanobe, Learning path recommender system based on recurrent neural network, in: Proceedings of 2018 9th International Conference on Awareness Science and Technology, IEEE, 2018, pp. 324–329.
[17] Tianyu Zhu, Zhenya Huang, Enhong Chen, Qiliu, Runze Wu, Le Wu, Yu Su, Zhigang Chen, Guoping Hu, Recommendation method for personalized test questions based on cognitive diagnosis, J. Comput. 40 (1) (2017) 176–191.
[18] Aurora Esteban, Amelia Zafra, Cristóbal Romero, Helping university students to choose elective courses by using a hybrid multi-criteria recommendation system with genetic optimization, Knowl.-Based Syst. 194 (2020) 105385.
[19] John K. Tarus, Zhendong Niu, Ghulam Mustafa, Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning, Artif. Intell. Rev. 50 (1) (2018) 21–48.
[20] Karim Dahdouh, Lahcen Oughdir, Ahmed Dakkak, Abdelali Ibriz, Smart courses recommender system for online learning platform, in: Proceedings of the 5th IEEE International Congress on Information Science and Technology, IEEE, 2018, pp. 328–333.
[21] Tuanji Gong, Xuanxia Yao, Deep exercise recommendation model, Int. J. Model. Optim. 9 (1) (2019) 18–23.
[22] Hao Wu, Zhengxin Zhang, Kun Yue, Binbin Zhang, Jun He, Liangchen Sun, Dual-regularized matrix factorization with deep neural networks for recommender systems, Knowl.-Based Syst. 145 (2018) 46–58.
[23] Zafar Ali, Guilin Qi, Pavlos Kefalas, Waheed Ahmad Abro, Bahadar Ali, A graph-based taxonomy of citation recommendation models, Artif. Intell. Rev. 53 (7) (2020) 5217–5260.
[24] Daqian Shi, Ting Wang, Hao Xing, Hao Xu, A learning path recommendation model based on a multidimensional knowledge graph framework for e-learning, Knowl.-Based Syst. 195 (2020) 105618.
[25] Wei Wang, Guangquan Zhang, Jie Lu, Member contribution-based group recommender system, Decis. Support Syst. 87 (2016) 80–93.
[26] Devendra Singh Chaplot, Eunhee Rhim, Jihie Kim, Predicting student attrition in moocs using sentiment analysis and neural networks, in: Proceedings of International Conference on Artificial Intelligence in Education, vol. 53, 2015, pp. 54–57.
[27] Tsung-Yen Yang, Christopher G. Brinton, Carlee Joe-Wong, Mung Chiang, Behavior-based grade prediction for MOOCs via time series neural networks, IEEE J. Sel. Top. Sign. Proces. 11 (5) (2017) 716–728.
[28] Michael V. Yudelson, Kenneth R. Koedinger, Geoffrey J. Gordon, Individualized bayesian knowledge tracing models, in: Proceedings of International Conference on Artificial Intelligence in Education, Springer, 2013, pp. 171–180.
[29] Albert T. Corbett, John R. Anderson, Knowledge tracing: Modeling the acquisition of procedural knowledge, User Model. User-Adapt. Interact. 4 (4) (1994) 253–278.
[30] Leonard E. Baum, Ted Petrie, Statistical inference for probabilistic functions of finite state Markov chains, Ann. Math. Stat. 37 (6) (1966) 1554–1563.
[31] Kai Zhang, Yiyu Yao, A three learning states Bayesian knowledge tracing model, Knowl.-Based Syst. 148 (2018) 189–201.
[32] Wei Tong, Fei Wang, Qi Liu, Enhong Chen, Data-driven mathematical test difficulty prediction, Comput. Res. Dev. 56 (5) (2019) 1007–1019.
[33] Justin C.W. Debuse, Victor J. Rayward-Smith, Feature subset selection within a simulated annealing data mining algorithm, J. Intell. Inf. Syst. 9 (1) (1997) 57–81.
[34] N.T. Heffernan, ASSISTments, 2014, https://sites.google.com/site/assistmentsdata/. (Accessed Dec. 29, 2019).
[35] J. Stamper, A. Niculescu-Mizil, S. Ritter, G.J. Gordon, K.R. Koedinger, Algebra 2005-2006. challenge data set from kdd cup 2010 educational data mining challenge, 2010, https://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp. (Accessed Dec. 29, 2019).
[36] Paul Steif, OLI engineering statics, 2011, https://pslcdatashop.web.cmu.edu/Project?id=48. (Accessed Dec. 29, 2019).
[37] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, Guangquan Zhang, Recommender system application developments: a survey, Decis. Support Syst. 74 (2015) 12–32.

[38] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, John Riedl, et al., Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International World Wide Web Conference, vol. 1, 2001, pp. 285–295.

[39] Ming Zhu, De-sheng Zhen, Ran Tao, You-Qun Shi, Xiang Yang Feng, Qian Wang, Top-n collaborative filtering recommendation algorithm based on knowledge graph embedding, in: Proceedings of the 14th International Conference of the Knowledge Management in Organizations, Springer, 2019, pp. 122–134.

[40] Guy Shani, Asela Gunawardana, Evaluating recommendation systems, in: Recommender Systems Handbook, Springer, 2011, pp. 257–297.

[41] Shah Khusro, Zafar Ali, Irfan Ullah, Recommender systems: Issues, challenges, and research opportunities, Inf. Sci. Appl. (2016) 1179–1189.

[42] Marius Kaminskas, Derek Bridge, Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems, ACM Trans. Interact. Intell. Syst. 7 (1) (2017) 2:1–2:42.

[43] Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, Supachanun Wanapu, Using of Jaccard coefficient for keywords similarity, in: Proceedings of the International Multiconference of Engineers and Computer Scientists, vol. 1, 2013, pp. 380–384.

[44] Dietmar Jannach, Markus Zanker, Alexander Felfernig, Gerhard Friedrich, Recommender Systems: An Introduction, Cambridge University Press, 2010.

[45] Jinjin Zhao, Shreyansh P. Bhatt, Candace Thille, Neelesh Gattani, Dawn Zimmaro, Cold start knowledge tracing with attentive neural turing machine, in: Inproceedings of the 7th ACM Conference on Learning @ Scale, Virtual Event, ACM, 2020, pp. 333–336.

[46] Thanawin Rakthanmanon, Bilson J.L. Campana, Abdullah Mueen, Gustavo E.A.P.A. Batista, M. Brandon Westover, Qiang Zhu, Jesin Zakaria, Eamonn J. Keogh, Searching and mining trillions of time series subsequences under dynamic time warping, in: Inproceedings of the 18th International Conference on Knowledge Discovery and Data Mining, ACM, 2012, pp. 262–270.

[47] Qian Zhang, Dianshuang Wu, Jie Lu, Feng Liu, Guangquan Zhang, A cross-domain recommender system with consistent information transfer, Decis. Support Syst. 104 (2017) 49–63.