Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 2157343, 13 pages
https://doi.org/10.1155/2021/2157343

*Research Article*

# Deep Learning and Collaborative Filtering-Based Methods for Students' Performance Prediction and Course Recommendation

**Jinyang Liu**[1], **Chuantao Yin**[2], **Yuhang Li**[2], **Honglu Sun**[2] and **Hong Zhou**[1]

[1]*School of Economics and Management, Beihang University, Beijing 100191, China*
[2]*Sino-French Engineer School, Beihang University, Beijing 100191, China*

Correspondence should be addressed to Chuantao Yin; chuantao.yin@buaa.edu.cn

Received 9 September 2021; Revised 25 October 2021; Accepted 3 November 2021; Published 2 December 2021

Academic Editor: Yinghui Ye



At the beginning of a new semester, due to the limited understanding of the new courses, it is difficult for students to make predictive choices about the courses of the current semester. In order to help students solve this problem, this paper proposed a hybrid prediction model based on deep learning and collaborative filtering. The proposed model can automatically generate personalized suggestions about courses in the next semester to assist students in course selection. The two important tasks of this study are course recommendation and student ranking prediction. First, we use a user-based collaborative filtering model to give a list of recommended courses by calculating the similarity between users. Then, for the courses in the list, we use a hybrid prediction model to predict the student's performance in each course, that is, ranking prediction. Finally, we will give a list of courses that the student is good at or not good at according to the predicted ranking of the courses. Our method is evaluated on students' data from two departments of our university. Through experiments, we compared the hybrid prediction model with other nonhybrid models and confirmed the good effect of our model. By using our model, students can refer to the different recommendation lists given and choose courses that they may be interested in and good at. The proposed method can be widely applied in Internet of Things and industrial vocational learning systems.

## 1. Introduction

The Internet of Things (IoT) is a huge network formed by combining all kinds of information sensing devices and networks to realize the interconnection of people, machines, and things anytime and anywhere. The rapid development of IoT has brought massive data support to machine learning, and the combination of IoT and deep learning methods will be the general trend in the future.

So far, there has been a lot of research on this. Huang et al. used a deep learning algorithm instead of manually monitoring the wearing of a safety helmet onsite [1]. Jiang et al. obtained semantic information of the scene by using the improved Faster-RCNN model [2]. Liao et al. used the improved SSD to carry out occlusion gesture recognition, realizing the interaction between machine and nature [3]. Gao et al. distinguish human left and right hands through deep convolution and feature extraction and also realize hand positioning and detection [4, 5]. In addition to these, the combination of IoT and deep learning can also help improve the efficiency of education systems. Mobile devices can collect data of students, and deep learning methods can be used to predict and explain students' progress and achievements. Deep learning can also be used for personalized recommendation modules to recommend more relevant content to educators. In this paper, we have carried out a thorough and detailed study on the last point.

In higher education, students will have many courses, including the required courses arranged by the university or the department and the elective courses that students can choose based on their needs. Reasonable choices of elective courses and being well-prepared for the courses of the coming semester can help a student to learn more and have better results. When choosing the elective courses, the university or the department will provide many choices for students, but

before studying these courses, students' understanding of these courses is limited, so it is hard for them to decide by themselves which courses are suitable for them. Our method combines deep learning methods and traditional methods to predict students' performances and interests based on history data. This method will provide each student several lists of courses which include the list of elective courses which match students' interests and which the student might be good at, the list of elective courses which match the students' interests and which the student might not be good at, and the lists of required courses which the student might be good at and which the student might not be good at.

Student performance prediction is always a major concern in the education domain. Many machine learning algorithms have been applied to predict students' performance in previous studies, like support vector machine [6], decision tree [7], linear regression [8], and random forest [9]. With the development of deep learning, many deep learning algorithms have achieved better performances than traditional machine learning algorithms on many different domains. Two domains that are most influenced by deep learning methods are computer vision [10] and natural language processing [11]. Basing on a large amount of image data and their associated labels, convolutional neural networks [12] can be trained to extract meaningful feature vectors from images, and these feature vectors can be further used for many different tasks, like classification [13] and object detection [14]. Basing on a large amount of text data, by deep learning algorithms, we can train a word embedding model, which projects each word into a vector in the latent space. The distance between different vectors in this latent space measures the semantic similarity between words. In both of these two domains, by using deep learning algorithms, the original data is transformed into latent representation (original input is projected to a vector in the latent space), which can be further used for other missions. So, it is important to study the application of deep learning algorithms in the education domain to see whether we can obtain the latent vectors of students and the latent vectors of courses basing on history data, which can measure the semantic similarity between students and between courses, respectively.

Experiments with different train/test data split modes show that the Course2Student algorithm can improve the accuracy of students' ranking prediction. Another reason for which we propose this new algorithm is to improve the interpretability of the model. When using models like neural collaborative filtering [15], each student and each course are associated with a latent vector, but the physical meanings of values in the latent vector of student and the latent vector, of course, are hard to explain. On the contrary, by using the Course2Student algorithm, even though the exact meanings of the obtained latent vectors are still hard to know, we can know the contribution of each associated course when predicting the ranking of a student on a new course, which can make us have more understanding of how the prediction result is made. On the other hand, some previous studies about student performance prediction only concentrate on one or several related courses [16], and the inputs of models are usually fixed, so to apply these methods to our scenario, multiple models with different inputs must be trained [17]. Moreover, our Course2Student is more flexible, and predictions can always be made no matter which courses a student has learned. And we only need one course embedding model for the prediction of all courses.

This study also compares the Course2Student algorithm with the nonparametric algorithm: user-based collaborative filtering [18]. To compare these algorithms, we also use different train/test data split modes. Experiment results show that Course2Student is better than user-based collaborative filtering on these data. To further improve the accuracy and reliability of the prediction result, we use the hybrid prediction method by combining the prediction result of Course2Student and the prediction result of user-based collaborative filtering [19]. Experiment results show that the hybrid prediction method can achieve higher accuracy than the single prediction method by selecting the prediction results with high confidence.

Most of course recommendation algorithms in the previous studies are commonly used recommendation algorithms, which can also be used in other domains like movie recommendation and product recommendation, and which are mainly based on collaborative filtering methods [20]. For the ranking prediction problem, Liu et al. proposed an improved probabilistic latent semantic analysis model (PLSA) [21] and a KNN-based optimal acceptance loss function Eigenrank [22]. Markus et al. proposed a model based on CofiRank-maximum margin matrix factorization (MMMF) technique [23]. Yue et al. proposed a model xCLiMF to optimize the ranking learning evaluation index MRR [24].

The main idea of collaborative filtering is to find the similarities between users or items and the relations between users and items, basing on the interaction history between users and items. These similarities and relations mainly describe the user's interests and the item's properties. In the course recommendation task, people should consider whether the course will match the student's interests and consider whether the student will be good at this course. In our course recommendation algorithm, we first select some courses that match students' interests by using a user-based collaborative filtering algorithm. Using the ranking prediction algorithm discussed above, the final recommendation lists are obtained, which not only consider students' interests but also consider their predicted performances.

The rest of this paper is organized as follows: Section 2 discusses related work, followed by the models for energy-efficient optimization and makespan optimization designed in Section 3. The improved clonal selection algorithm for resource allocation is discussed in Section 4. Section 5 shows the simulation experimental results, and Section 6 concludes the paper with summary and future research directions.

## 2. Related Works

This section will talk about some existing works that are most related to our works, including neural networks [25], collaborative filtering, and neural network-based collaborative filtering [26].

*2.1. Neural Networks.* Neural networks are firstly inspired by neural science, and their initial objective is to simulate how

information transfers in the human brain. Neural networks consist of many connections and nodes. Information runs through these connections and nodes. In fact, most of the neural networks used in today's field of research can be regarded as a collection of many linear functions and nonlinear activation functions; for example, logistic regression can be regarded as an example of the simplest neural networks, which consists of one linear function and one nonlinear activation function (sigmoid).

Neural networks can also be regarded as a collection of many layers. These layers can be classified into three groups: input layer, hidden layer, and output layer. Information transfers through neural networks with a fixed direction. Neural networks can be used as an automatic solution for many tasks. Taking the classification problem as an example, after the original image being fed into neural networks, based on the characteristic of the input image, different neurons will be activated in different layers; outputs of deeper layers have more information for the classification mission. Normally, the number of neurons in the output layer is the same as the number of categories. The output value of each neuron in the output layer describes the predicted probability of the associated category.

In order to obtain a neural network for a specific task, we need to use data to train the neural network. When training a neural network, we need to provide the input and the output of the neural network, and the weights are optimized by the back propagation algorithm [27]. In fact, the main objective of a neural network is to simulate a function. Normally, this function is complicated, and it cannot be constructed by human analysis. However, it can be simulated by neural networks when given the input and output data.

Some previous works have used neural networks to predict student performance. In a very early study [28], neural networks are used to predict academic success in MBA programs. In this work, neural networks are compared with four other prediction methods: least square regression [29], stepwise regression, discriminate analysis [30], and logistic regression [31]. The object of this study is to help make the decision to accept students into the MBA program. In [32], an intelligent tutoring system based on neural networks is proposed. In order to provide an appropriate problem for a student, a neural network is trained at first to predict the number of errors that the student might make on a certain set of problems. Then, based on the prediction result, a suitable problem is decided for the student. Lykourentzou et al. used the students' prediction problem in an e-learning scenario [33]. The final grades are estimated based on the data collected before the middle of the course by a neural network-based model. Then, based on the predicted level of performance, students are clustered into two groups, and each group will be provided with suitable educational materials. One similar work [34] uses a neural network-based model to learn the interaction between students and courses to predict student performance.

*2.2. Collaborative Filtering.* The collaborative filtering algorithm [20] is the most used algorithm in the studies of recommendation system, and it has already been used in some real-life applications [35]. It was first introduced to recommend electronic documents to users [36]. The data, on which the collaborative filtering algorithm can be applied, can normally be represented by an interaction matrix which describes the interaction between the users and the items. In this interaction matrix, each row presents a user and each column presents an item; the values in this matrix presents the interaction between the related user and the related item. This interaction matrix is always sparse because many interactions between users and items are unknown. The main task of the collaborative filtering algorithm is to predict the unknown interactions basing on the existing interactions and, basing on the prediction results, make recommendations for users. In fact, the collaborative filtering algorithm is a collection of many algorithms. Most of today's collaborative filtering algorithms can be classified into two groups: similarity-based algorithm and latent factor algorithm.

There are two kinds of similarity-based algorithms: item-based collaborative filtering [20] and user-based collaborative filtering [37]. A similarity-based algorithm is a very intuitive algorithm, and it is mainly based on the similarities between users or the similarities between items. The similarities between users can be obtained by calculating the similarities between rows in the interaction matrix, since each row presents the interaction between the current user and all the items. Similarly, the similarities between items can be obtained by calculating the similarities between columns in the interaction matrix. A user-based collaborative filtering algorithm can be presented by

$$v_{a,j} = \sum_{i \in \text{neighbors}(a)} w_i v_{i,j}. \tag{1}$$

Here, we want to predict the interaction between user $a$ and item $j$. The prediction is based on the interactions between item $j$ and some users similar to user $a$. In this equation, $w_i$ is proportional to the similarity between user $I$ and current user $a$, which means that the users who are more similar to current user $a$ will have more contribution on the prediction result.

The latent factor algorithm is an improvement of the content-based algorithm [38]. In the latent factor algorithm, we suppose that each user and each item have a latent representation. These latent representations can be used to describe the interaction between users and items. The recommendation is based on the similarity between the user's latent vector and the item's latent vector. Different from a content-based algorithm where the features of users and items are decided by human experts, in the latent factor algorithm, the latent vectors depend on history interactions between users and items and the interaction function, which measures the similarity between two latent vectors. One example of a latent factor algorithm is the matrix factorization method [35], as shown in equation (2). Here, the interaction between a user and an item is represented by the inner product of the user's latent vector and the item's latent vector. Many studies have focused on how to improve the latent representation and how to improve the interaction function:

$$v_{a,j} = v_a^T v_j, \tag{2}$$

where $v_{a,j}$ is the interaction between a user and an item, $v_a$ the user's latent vector and $v_j$ the item's latent vector.

Some previous works have applied a collaborative filtering algorithm to the course recommendation problem. In [39], collaborative filtering is combined with the Artificial Immune System. Students are first placed into several clusters using the Artificial Immune System clustering approach to calculate the affinities between different students in a training data pool. Then, collaborative filtering is applied to the data cluster to predict the rating for the course. In [40], collaborative filtering is combined with students' online learning style. Basing on their online learning styles, students are first clustered by the $k$-means algorithm. Then, item-based collaborative filtering algorithms and user-based collaborative filtering algorithms are applied to each cluster. In [41], a matrix factorization-based method is proposed to predict students' feedback ratings on courses. This work targets three problems: potential lack of rating data from students to courses, imbalance of the user-item matrix, and dependencies between courses.

There are also some works which use the collaborative filtering algorithm for students' performance prediction. In [42], both user-based collaborative filtering algorithms and item-based collaborative algorithms predict students' grades on elective courses. The objective of this work is to recommend elective courses for each student on which the student might have higher grades. Their experiments prove that the performances of user-based collaborative filtering algorithms and item-based collaborative filtering algorithms are similar in their data. The idea of this study is similar to the idea of our work. In a more recent work [43], a novel cross-user-domain collaborative filtering algorithm is designed to accurately predict the score of the optional course for each student by using the course score distribution of the most similar senior students and recommend the top $t$ optional courses with the highest scores without time conflict. The difference is that, in our work, we consider not only the predicted performances of the student but also their interests in these courses. Based on this work's results, we also use a user-based collaborative filtering algorithm as one of our baseline methods for student performance prediction.

*2.3. Neural Network-Based Collaborative Filtering.* In fact, the neural network-based collaborative filtering algorithm [22] is a kind of latent factor algorithm, and it is also a popular direction of research in recent years [44]. The advantages of the neural network-based collaborative filtering algorithm is that its interaction function is based on neural networks which can be learned from the data, while in the previous algorithms, the interaction function is decided by a human, so if the chosen interaction function is not suitable for the current dataset, then the algorithm's performance will decrease. One of the most important neural network-based collaborative filtering works is neural collaborative filtering [22]. In this algorithm, each user and each item have two latent vectors. When calculating the interaction between the user's latent vectors and the item's latent vectors, the first result is obtained by the element-wise product between the first latent vector of the user and the first latent vector of the item. To get the second result, the second latent vector of the user and the second latent vector of the item are concatenated and used as input of a neural network, and the output of this neural network gives us the second result. The first and the second results are then concatenated, and another neural network is applied to this concatenated vector to get the final prediction of the interaction between the user and the item. The reason for using two latent vectors is that in this way, the interaction function can have both linear and nonlinear parts.

Few works have used neural collaborative filtering to predict the interactions between students and courses. In most recent works, Sun et al. used a multitask learning strategy to improve the neural collaborative filtering method and use it to predict student performance, and [45] used the instructor's identity as another input of the neural collaborative filtering model. The main idea of all these recent similar works is to use a neural network to the latent factor collaborative filtering method. Different from these works, in our work, we combine neural collaborative filtering with traditional similar-based collaborative filtering methods in order to make the prediction process more reasonable. And since neural collaborative filtering is the most used one in the previous works, we use it as one baseline method.

## 3. Course Recommendation Method

Our overall model is shown in Figure 1. It is mainly divided into two parts, namely, course recommendation and student ranking prediction. In the third and fourth sections, we will introduce the model design of these two parts, respectively.

The user-based collaborative filtering method is used to predict the elective courses which might match students' interests. The final recommendation list is a combination of the prediction of students' interests and the prediction of students' performance. The method for the prediction of students' performance is discussed in the previous section. In the prediction scenario of students' interest, we suppose that the students are from different years, noted as $Y_1, Y_2, \cdots, Y_m$ $(Y_1 < Y_2 < \cdots < Y_m)$. A student of year $Y_1$ means that the student begins his (her) study at university at year $Y_1$. To make recommendation for a target student of year $Y_k$ and for courses in semester $q$, there are two steps: selection of similar students and selection of recommended courses.

The final output of the model is three lists. List 1 is the list of recommended courses that the student might be good at; lists 2 and 3 are the lists of recommended courses and required courses that the student might not be good at. $Y_i S_m$ is the Top-$m$ most similar student from year $Y_i$. $Y_i S_m$ sim is the similarity between $Y_i S_m$ and the target student. $Y_i S_j C_k$ represents a course learned by $Y_i S_j$ but is not learned by the target student.

*3.1. Selection of Students.* Basing on the previous selected courses of the first $q-1$ semesters, we select $N_0$ most similar students from the students of each previous year

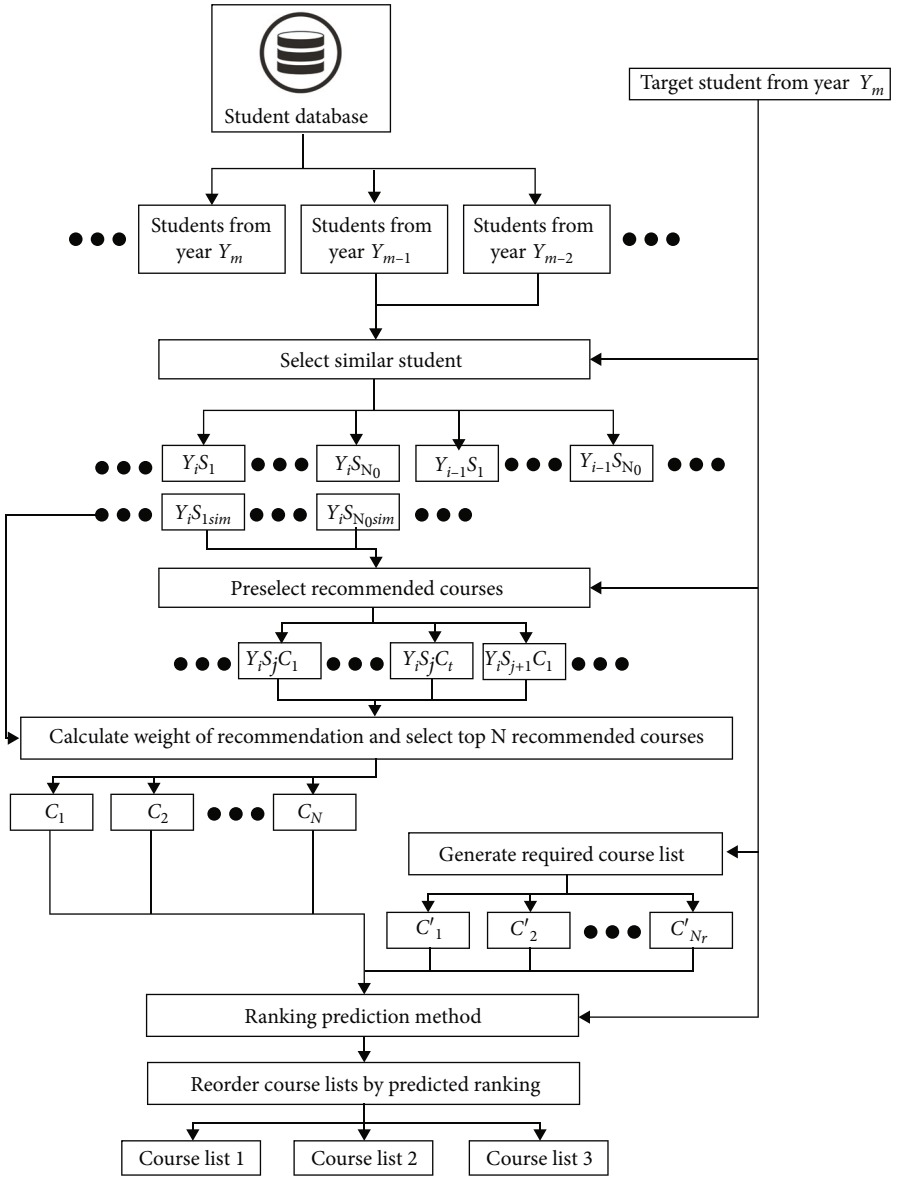


Figure 1: Visualization of the system model.

$(Y_1, Y_2, \cdots, Y_{k-1})$. The method to calculate the similarity is presented at the end of this part. These selected students are presented by set $\{Y_iS_j, i \in [1, k-1], j \in [1, N_0]\}$ which is named as the set of similar students. Meanwhile, each student in the set of similar student has a value of similarity compared to the target student; these values of similarities are presented by the set $\{Y_iS_j\text{sim}, i \in [1, k-1], j \in [1, N_0]\}$ in which $Y_iS_j\text{sim}$ presents the similarity between student $Y_iS_j$ and the target student.

*3.2. Selection of Recommended Courses.* After obtaining the set of similar students, for each student in this set, we select the courses which he (she) has learned in the first $q$ semesters and which the target student has not learned in the first $q-1$ semesters. These courses are presented by set $\{Y_iS_jC_t, i \in [1, k-1], j \in [1, N_0], t \in [1, Y_iS_jN]\}$ in which $Y_iS_jN$ present the number of courses which student $Y_iS_j$ has learned and the target student has not learned. In fact, in this set, different elements may be associated with the same course. This set is named the set of preselected courses. Then, we calculate the weight of recommendation of each course that appears in the set of preselected courses as shown in equation (3). In this equation, if $\text{course}_{\text{name}}$ is the same as $Y_iS_jC_t$, then the function $I(\text{course}_{\text{name}} = Y_iS_jC_t)$ will return to 1; otherwise, it will return to 0. The weight of recommendation describes quantitatively whether the course should be recommended, and this value can be used to sort the top $N$ recommendation list:

$$\text{weight}_{\text{rec}}(\text{course}_{\text{name}}) = \sum_i \sum_j \sum_t Y_iS_j\text{sim} * I(\text{course}_{\text{name}} = Y_iS_jC_t). \tag{3}$$

In the next part, the method to calculate the similarity between courses and the method to measure the importance

of course are presented, in which the importance of course is an important indicator when calculating the similarity between courses.

*3.3. Measure the Importance of Courses.* Breese et al. proposed in [46]: similar ratings on some popular items do not represent a good indication that the two users have similar preferences, and similar ratings on niche items are more meaningful for reference.

The same idea can be used to optimize our model: The influences of different courses on the description of students' personality are different. The courses which are chosen by fewer students can better describe their personalities. The courses which are chosen by most of the students are usually some necessary courses for their major, and so these courses cannot describe students' personalities. Basing on this idea, we use equation (4) to measure the importance of the course. Therefore, we will reduce the weight of popular courses and increase the weight of minority courses. Here, $N_{\text{total}}$ presents the total number of students in the current department and $N_{\text{course}}$ presents the number of students who have chosen the current course:

$$\text{weight}(\text{course}) = \frac{N_{\text{total}}}{N_{\text{course}}}. \tag{4}$$

*3.4. Measure the Similarity between Students.* According to our method, the similarity between student $S_i$ and student $S_j$ is the similarity between the set of courses learned by student $S_i$: $\{C_{i,k}\}$ and the set of courses learned by student $S_j$: $\{C_{j,k}\}$. The method is shown in

$$\text{sim}(S_i, S_j) = \frac{\sum_{c \in \{C_{i,k}\} \cap \{C_{j,k}\}} \text{weight}(c)}{\sum_{c \in \{C_{i,k}\} \cup \{C_{j,k}\}} \text{weight}(c)}. \tag{5}$$

## 4. Ranking Prediction Method

There are many ways to describe a student's performance on a course. Two mainly used ways are students' score and ranking. A student's score can be influenced by many factors. Different courses may have different means and variances. The same course may also use different ways for evaluation on different semesters. Also, the score is uncertain. Even though the observed score is a fixed number, the ground truth score could be a distribution. On the contrary, students' ranking is a better choice for prediction. Firstly, the ranking will not be influenced by the mean and the variance of scores. Secondly, ranking can be used as an indicator that can directly describe whether the student is good at this course or not. Meanwhile, to better decrease the influence of uncertainty on the prediction results, we regroup the students into two categories. The first category contains students whose rankings are under 50%. The second category contains the students whose rankings are above 50%. In this way, the uncertainty of the score can only influence the students whose rankings are near 50%.

*4.1. Course2Student.* Our proposed method is named as Course2Student since our main idea is to use the characteristic of the previous courses and the associated performances to describe the characteristic of the student. Before introducing our method, we will first make a summary of the existing methods for students' performance prediction.

Most of the methods for student performance prediction can be regarded as a function. The inputs of the function are some indicators of students which are correlated with students' performance. The output is the students' predicted performance. These methods can be divided into two types: parametric method and nonparametric method. For the nonparametric method, a rule is proposed. Basing on this rule and the inputs of the prediction model, the correlated information is selected from the data, and then, they are used to predict the final result. Some commonly used nonparametric methods include $K$-nearest neighborhood and collaborative filtering [20]. For parametric methods, there will be some parameters in the model, and the data optimize these parameters. We can consider that the nonparametric model directly takes data as memories and the parametric model first learn something from the data and then forget about the original data.

Course2Student is a parametric method. It is based on neural collaborative filtering and linear regression. Linear regression is one of the earliest methods used for students' performance prediction [47], and even in recent years, it is still a commonly used method [48]. Equation (6) is an example of linear regression, in which $S$ presents the student's performance which we want to predict, $X_i$ presents one indicator of students which can be used to predict the student's performance, and $w_i$ presents the associated weight of indicator $X_i$. In previous works, many indicators have been studied for the prediction of student performance [48] Generally, the most correlated indicators to students' performance prediction are the students' performances on his (her) previous courses. So, in our work, we choose students' performances on his (her) previous courses as indicators for performance prediction:

$$S = \sum_i w_i X_i. \tag{6}$$

One advantage of linear regression is that based on the weights associated with each indicator, we can easily understand the importance of each indicator on the prediction result. One disadvantage of linear regression is that, after training, the model can only be used in the current mission, which means that it can only be used to predict the performance of one course. In a real-life application, we need to predict the results of many courses, so we have to train separately many independent linear regression models. Also, since the weights are correlated with the indicators, if one indicator in a model is changed, then we have to train a new model. To solve the above problems, one possible solution is to reuse the weights, as shown in equation (7). Since in our method the rankings of new courses are predicted by previous courses' rankings, we use $X$ to present both the prediction results and the indicators. In equation (7), $X_j$ presents the ranking of the current student on course $j$, $w_{i,j}$ describes the influence of course $i$ on the ranking prediction of course $j$, or we can say that it describes the

similarity between course $i$ and course $j$ on the ranking prediction problem. Set $A$ presents the set of courses that the current student has learned. By this method, after obtaining all the weights in $\{w_{i,j}\}$, given one student and one course, the student's ranking on that course can be predicted basing on his (her) rankings of previous courses:

$$X_j = \sum_{i \in A} w_{i,j} X_i. \tag{7}$$

The above solution makes the prediction model more flexible, and it still could be further improved. By this method, supposing that there are $N$ courses, then $\{w_{i,j}\}$ will have $N^2$ parameters (supposing that the similarity between two courses is not symmetric which means that $w_{i,j}$ is not necessarily equal to $w_{j,i}$ when $i \neq j$). But in fact most of the weights in $\{w_{i,j}\}$ are not independent, so it is not necessary to have $N^2$ parameters. For example, if the ranking of course $k$ only depends on the ranking of course $i$ as $X_k = w_{i,k} X_i$, and the ranking of course $j$ only depends on the ranking of course $i$ as $X_j = w_{i,j} X_i$, then when predicting the ranking of course $k$ basing on the ranking of course $j$, we will have $X_k = w_{j,k} X_j = w_{i,k}/w_{i,j} X_j$, which means that $w_{j,k} = w_{i,k}/w_{i,j}$; these three parameters are not independent. In order to further improve the prediction method, we propose a new method as shown in equation (8), which is our proposed Course2Student method. In this equation, embedding$(\cdot)$ is a function which projects the identity of the course to the latent vector of the course. sim$(\cdot)$ is a function basing on neural networks which measures the similarity between two latent vectors of courses in ranking prediction problems:

$$X_j = \sum_{i \in A} \text{sim}(\text{embedding}(i), \text{embedding}(j)) X_i. \tag{8}$$

The function embedding$(\cdot)$ has $MN$ parameters where $M$ presents the length of the latent vector of the course. Since there are relatively many courses $M < N$, this method has fewer parameters compared with the previously discussed method. Besides, this method can keep the transitivity of similarity, which means that if course $i$ is similar to course $j$ and course $j$ is similar to course $k$, then by using the Course2Student method, we can obtain that course $i$ is also similar to course $k$, which is logical. The idea of using neural networks to measure the similarity comes mainly from the work of neural collaborative filtering. Since it is hard for human to choose the best way of the measure of similarity basing on the given data, it is better to use neural networks to learn a measure of similarity directly from the data. The Course2Student method is also illustrated in Figure 2.

*4.2. Selection of Correlated Courses.* In our prediction scenario, normally, a student will have a lot of previous courses; it will take a lot of time if we use all these previous courses to train the Course2Student model. So, before training the Course2Student model, for each target course for prediction, $N_0$ most correlated courses are selected. The correlation between two courses is calculated by Pearson's correlation coefficient, as shown in equation (9), in which the correlation between course $x$ and course $y$ is calculated. In this equation, $n$ presents the number of students who have learned both course $x$ and course $y$, and $x_i$ and $y_i$ present, respectively, the score on course $x$ and the score on course $y$ of the $i^{\text{th}}$ student who has learned both course $x$ and course $y$. Considering that some courses might have very few learners, in that case, Pearson's correlation coefficient may not be able to describe the correlation between these courses properly. Therefore, when calculating the correlation, if the number of students who have learned both courses is under 30, then the correlation between these two courses is considered 0:

$$\text{correlation}(x, y) = \frac{\sum_i^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i^n (x_i - \bar{x})^2} \sqrt{\sum_i^n (y_i - \bar{y})^2}}. \tag{9}$$

*4.3. Hybrid Prediction Method.* Considering that different prediction results have different uncertainties, but one determinist prediction model can only give one result, basing on this result, it is hard to measure the uncertainty of this prediction. In order to better estimate the uncertainty of prediction and filter out the uncertain prediction results to improve the overall accuracy, we use a hybrid prediction method by combining Course2Student and a user-based collaborative filtering method.

We choose these two methods because Course2Student is a parametric method and user-based collaborative filtering is a nonparametric method, so these two methods will have very different decision boundaries. If these methods give the same prediction result, then we consider that this prediction result has high confidence. In the hybrid prediction method, we only consider the result which has high confidence as an available prediction result. Basing on this hybrid method, we can divide each course list (list of required courses or list of courses that the student might like) into two sublists. The first sublist is the list of courses that the student might be good at (available ranking prediction result is in top 50%), and the second sublist is the list of courses that the student might not be good at (available ranking prediction result is not in 50%).

The user-based collaborative filtering method for ranking prediction is shown in equation (10). In order to predict the ranking of student $a$ on course $j$ which is presented by $X_{a,j}$, we first select a set of students who are most similar to student $a$ and who have also learned course $j$, then use their rankings on course $j$ and their similarities with student $a$ to predict $X_{a,j}$. Function sim$(i, a)$ measures the similarity between two students, $C(i)$ presents the set of courses which student $i$ has learned, and card $(A)$ presents the number of elements in set $A$:

$$X_{a,j} = \sum_{i \in \text{neighbors}(a)} \text{sim}(i, a) X_{i,j},$$

$$\text{sim}(i, a) = \frac{1}{\sum_{j \in C(i) \cap C(a)} \left(X_{i,j} - X_{a,j}\right)^2 / \text{card}\,(C(i) \cap C(a))}. \tag{10}$$

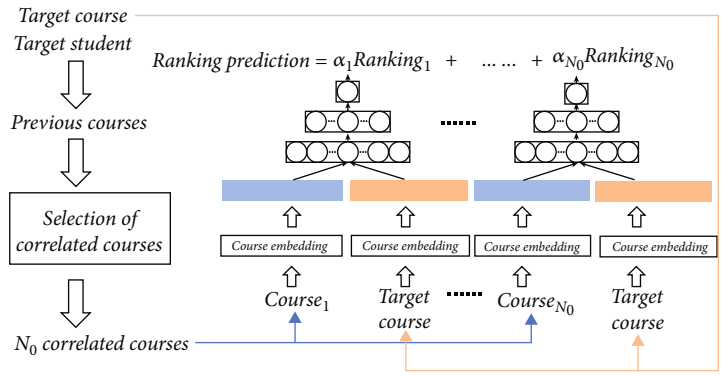


Figure 2: Course2Student ranking prediction method.

Table 1: Recommendation results.

| Method | User-based collaborative filtering | | | |
|---|---|---|---|---|
| Department | Computer science | | | |
| Year | 2016 | | | 2017 |
| Semester | 5 | 7 | 8 | 5 |
| Recall | | | | |
| $N = 10$ | 0.1164 | 0.1421 | 0.1392 | 0.1711 |
| $N = 20$ | 0.1674 | 0.196 | 0.2187 | 0.3061 |
| $N = 30$ | 0.2179 | 0.2342 | 0.2888 | 0.3809 |
| Precision | | | | |
| $N = 10$ | 0.0269 | 0.0484 | 0.0329 | 0.0432 |
| $N = 20$ | 0.0182 | 0.0324 | 0.0262 | 0.0365 |
| $N = 30$ | 0.0161 | 0.0258 | 0.0233 | 0.0303 |
| Department | Honors college | | | |
| Year | 2016 | | | 2017 |
| Semester | 5 | 7 | 8 | 5 |
| Recall | | | | |
| $N = 10$ | 0.539 | 0.4637 | 0.3977 | 0.4667 |
| $N = 20$ | 0.6288 | 0.553 | 0.5208 | 0.5289 |
| $N = 30$ | 0.6741 | 0.5912 | 0.5775 | 0.568 |
| Precision | | | | |
| $N = 10$ | 0.2957 | 0.3657 | 0.2862 | 0.2796 |
| $N = 20$ | 0.1755 | 0.2234 | 0.1885 | 0.1598 |
| $N = 30$ | 0.1374 | 0.1634 | 0.1404 | 0.1155 |

Table 2: Train/test data split options.

| Year | Computer science and honors college | | | | Data split 1 | Data split 2 | Data split 3 |
|---|---|---|---|---|---|---|---|
| | 2015 | 2016 | 2017 | 2018 | | | |
| Semester | 1 | | | | Train | Train | Train |
| | 2 | | | | | | |
| | 3 | | | | | | |
| | 4 | 1 | | | | | |
| | 5 | 2 | | | | | |
| | 6 | 3 | | | | | |
| | 7 | 4 | 1 | | | | Test |
| | 8 | 5 | 2 | | | | |
| | 9 | 6 | 3 | | | | |
| | 10 | 7 | 4 | 1 | Test | Test | |
| | 11 | 8 | 5 | 2 | | | |
| | 12 | 9 | 6 | 3 | | | |

## 5. Experiment

In order to evaluate our methods, we collect our own dataset. This dataset contains students of years 2015, 2016, 2017, and 2018 from the department of computer science and honors college at our university. The data of the department of computer science contains 844 students and 715 courses. The data of honors college contains 977 students and 1457 courses. According to the education system of our university, a bachelor's degree takes 4 years. Each year has three semesters, the semester of autumn, the semester of spring, and the semester of summer. The semester of summer is a short semester, and only a part of the students will take this semester. So before obtaining the bachelor's degree, a student will have totally 12 semesters. Until these data are collected, we get data of 12 semesters for the students 2015, 9 semesters for the students 2016, 6 semesters for the students 2017, and 3 semesters for the students 2018. In order to protect students' privacy, the identity of each student is replaced by a unique string. In the original data, there is no information about whether the course is required or elective. On the other hand, as the students from different years might have different required course lists, for now, we have not got all the required course lists for all students. In order to know whether a course is an elective course or not, we check the number of students who have learned the course. If more than half of the students have learned the course, then the course will be treated as a required course. Otherwise, it will be treated as an elective course.

*5.1. Advice on Equation Results of Course Recommendation.* In this part of the experiments, we make a recommendation for the courses of semesters 5, 7, and 8 of the students of year

Table 3: Prediction results of student ranking.

| Method | Department | Option | Accuracy |
|---|---|---|---|
| Neural collaborative filtering [26] | Computer science | 1 | 0.6765 |
| | | 2 | 0.6458 |
| | | 3 | 0.6391 |
| | Honors college | 1 | 0.7025 |
| | | 2 | 0.7055 |
| | | 3 | 0.6895 |
| User-based collaborative filtering | Computer science | 1 | 0.6751 |
| | | 2 | 0.6581 |
| | | 3 | 0.6609 |
| | Honors college | 1 | 0.7116 |
| | | 2 | 0.7214 |
| | | 3 | 0.6949 |
| Course2Student | Computer science | 1 | **0.6938** |
| | | 2 | **0.6635** |
| | | 3 | **0.6616** |
| | Honors college | 1 | **0.7208** |
| | | 2 | **0.7250** |
| | | 3 | **0.7088** |

2016 and for the courses of semester 5 of the students of year 2017. The reason why we choose these semesters is that there are relatively more elective courses, so it is better for the evaluation of recommendation methods. The recommendation lists are generated based on the previous choices of courses. We use recall and precision to evaluate the recommendation results, and different lengths of recommendation lists are used: 10, 20, and 30. Table 1 shows the results by using the user-based collaborative filtering method. Here, we are more interested in the results of recalls because in this mission, we want that the recommendation list can contain all the courses that the student might choose and at the same time provide some courses that the student has not noticed but may catch his (her) interests.

Precision is also used since it is commonly used to evaluate the recommendation results. We can see that the results on honors college are much better than the results on the department of computer science. It may be caused by the fact that the students from the honors college will choose their options at the end of the second year, so from their choices of elective courses, we can see some personal characteristics. From this point of view, we can see that this recommendation method is more suitable for the recommendation scenarios where students have more elective courses to choose from and where students have their own options to choose from. Considering the values of recall, the average value of recalls of semester 5 of the students from the honors college of year 2016 is above 0.5 even though $N = 10$, which is a result that could prove that the current method can be used in real life. In future work, we will pay more attention on the departments which education systems are similar to the education system of the honors college and explore new recommendation methods for the departments in which students have relatively fewer elective courses. For example, we can combine natural language processing methods or knowledge graph [42] to add some prior knowledge about the courses.

Table 4: Prediction results of student ranking by hybrid method.

| Method | Department | Option | Accuracy | Coverage |
|---|---|---|---|---|
| Hybrid | Computer science | 1 | 0.7213 | 0.8219 |
| | | 2 | 0.7109 | 0.7775 |
| | | 3 | 0.7155 | 0.7478 |
| | Honors college | 1 | 0.7585 | 0.8344 |
| | | 2 | 0.7618 | 0.8551 |
| | | 3 | 0.7459 | 0.8225 |

*5.2. Results of Ranking Prediction.* In order to make our experiments more similar to the real prediction scenarios, we use the time node to separate the training and testing data. For example, for the students of year 2018, in order to get the ranking prediction after the second semester, we only use the data which can be collected before the beginning of the second semester. To better evaluate the generalization of our methods, we use three different time nodes to separate the training and testing data; for a student of year 2016, it is the time nodes before 8th, 7th, and 5th semesters. The details about the training/testing data split are shown in Table 2. In the second and third data split options, since all the data of students of year 2018 are in the testing data, we only consider the data of students of years 2015, 2016, and 2017.

We treat the students' ranking prediction problem as a binary classification problem. The rankings which are in the

Table 5: Prediction results of student ranking.

| Method | Department | Option | Ranking improvement | Improvement percentage |
|---|---|---|---|---|
| Hybrid | Computer science | 1 | 0.1677 | 0.7894 |
| | | 2 | 0.1011 | 0.7739 |
| | | 3 | 0.1537 | 0.7672 |
| | Honors college | 1 | 0.2138 | 0.7468 |
| | | 2 | 0.2307 | 0.7318 |
| | | 3 | 0.2405 | 0.7864 |

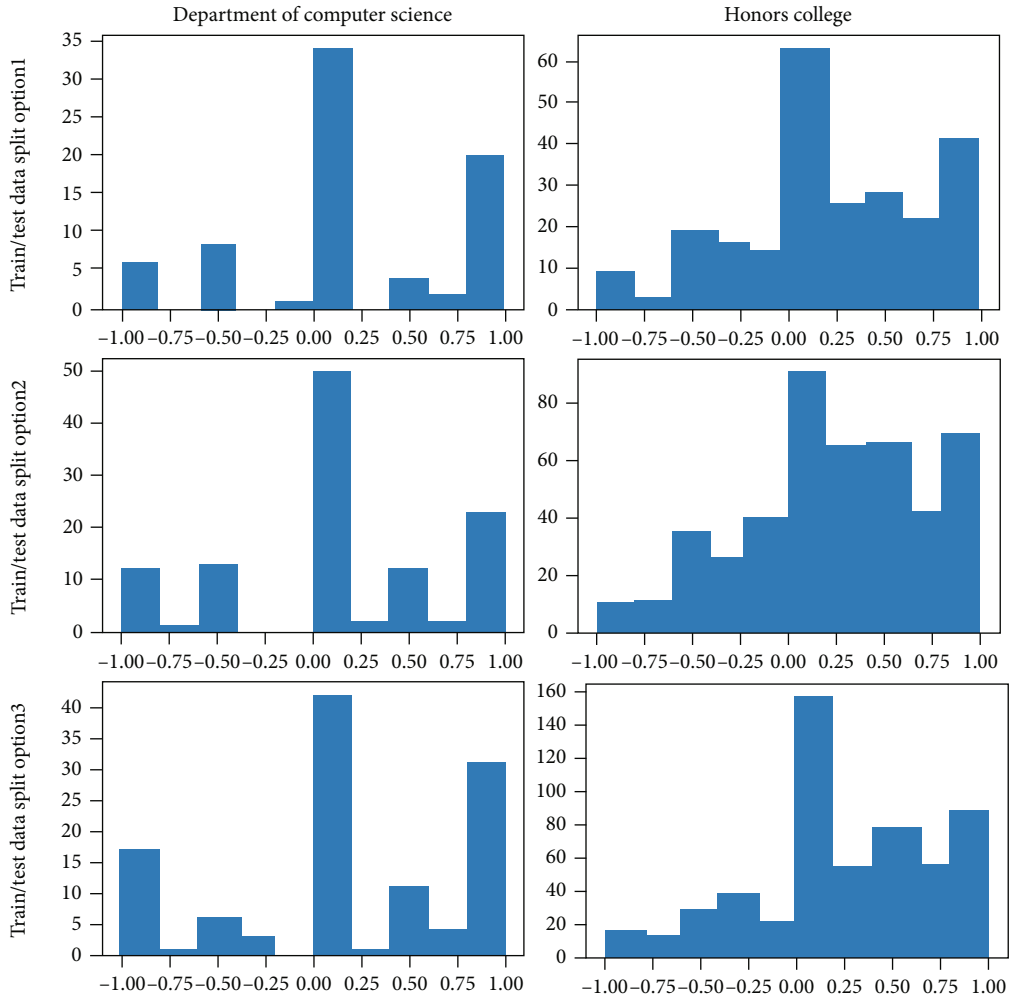


Figure 3: Distribution of ranking improvement.

first 50% are regarded as the first category, and the rankings which are not in the first 50% are regarded as the second category. In order to evaluate our proposed Course2Student method, its performance is compared with the performances of two other methods: neural collaborative filtering and user-based collaborative filtering, which are the two most used methods for similar problems. The accuracy of the testing data is used to evaluate the performance of each method. Table 3 shows the results.

For classification model $f$ and test data set $D$ with size $n$, accuracy is defined as

$$\text{Accuracy}(f; D) = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) = \text{label}_i). \tag{11}$$

We bold the highest accuracy of each data split option.

Results show that our method achieves the highest accuracies on all data and all training/testing split options.

To further improve the accuracy and the confidence of prediction results, we use the hybrid method. The prediction results of user-based collaborative filtering and Course2Student are combined. Only the same prediction results are regarded as available prediction results and used. The other results are regarded as results with high uncertainties. Table 4 shows the results of the hybrid method, which includes the accuracies and the percentages of the available results. We can see that the hybrid method can indeed further improve prediction accuracy, which means that this estimation of uncertainty is logical.

*5.3. Results of Ranking Prediction.* To evaluate whether this method can improve students' ranking, based on the results of the hybrid prediction method, we separate the list of elective courses into two sublists for each student in the testing data. The first sublist contains the courses which have a predicted ranking in the first 50%, and the second sublist contains the courses which have a predicted ranking not in the first 50%. For each student, we use the difference between the mean of the real rankings of the courses in the first sublist and the second sublist to describe the ranking improvement of that student by only choosing the elective courses that the student is predicted to be good at. Here, we use 0 and 1 to present the ranking. 1 means that it is in the top 50%, and 0 means that it is not in the top 50%. The ranking improvement is described by the mean of ranking improvements of all associated students. Table 5 shows the results. We can see that under different training and testing split options, there are always over 70% of students whose rankings are improved according to this method of evaluation. Figure 3 presents the details about the distribution of ranking improvement. One point that needs to be discussed is that now we only use the history data to evaluate our methods, but when these methods are applied in a real-life situation, will the result of the student be changed when he (she) knows the predicted result? In our future studies and applications, we will concentrate on this point.

*5.4. Advantages of the Proposed Method.* Our proposed method has four major advantages.

Firstly, the framework of the proposed ranking prediction method is based on the traditional method so that it can always have acceptable prediction results; on the contrary, when using the end-to-end machine learning methods, like the ones used in previous works, sometimes we could get some extremely abnormal result; for example, the predicted value can be out of the interval of possible results.

Secondly, the proposed ranking prediction method makes it possible for us to know the influences of each historical course on the ranking prediction of the target course, which can help us understand how the prediction result is made and as a result has a better vision of the resulting model. And this vision and understanding of the model can help us debug the model during the training process much easier compared with an end-to-end model.

Thirdly, as the recommended courses are firstly selected according to the students' preference and then reordered by the ranking prediction result, all the recommended courses are acceptable considering the student's interest. Some previous works use the performance prediction results directly to recommend courses, and in that way, some recommended results may be totally irrelevant to students' future plans.

Fourthly, when recommending courses only based on the performance prediction result, we need to predict students' performance on each course in the database, which is an extremely time-consuming process. In our method, it is only necessary to predict the performance of the student on the courses in the preselected list (selected by students' preference) which takes much less time.

## 6. Conclusions

In this work, we propose a new method that can automatically generate personal advice about courses in the next semester. Particularly, we explore the application of deep learning methods on students' ranking prediction problem and propose a new method that combines neural networks with traditional methods. The results of experiments prove that our methods can indeed recommend courses for students that can match their interests, and students have a high possibility to improve their average rankings when they choose the elective courses which, according to the prediction result, they might be good at. For now, our studies are mainly based on the existing data. In future work, we will concentrate more on the changes in students' behaviors when they know the prediction results. For example, students will change their learning strategies and achieve higher scores when they have known that they might not be good at certain courses. The relevant machine learning methods and course recommendation methods mentioned in this paper can be widely applied in IOT and industrial vocational learning systems. So in our next steps, we will work with the associated departments and try to apply our methods to a real-life system.

## Data Availability

The simulation experiment data used to support the findings of this study are available from the corresponding author upon request.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This work was supported by the National Key R&D Program of China (No. 2019YFB2102200) and National Science Foundation of China (Grant No. 61977003).

## References

[1] L. Huang, Q. Fu, M. He, D. Jiang, and Z. Hao, "Detection algorithm of safety helmet wearing based on deep learning," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 13, article e6234, 2021.

[2] D. Jiang, G. Li, C. Tan, L. Huang, Y. Sun, and J. Kong, "Semantic segmentation for multiscale target based on object recognition using the improved Faster-RCNN model," *Future Generation Computer Systems*, vol. 123, pp. 94–104, 2021.

[3] S. Liao, G. Li, H. Wu et al., "Occlusion gesture recognition based on improvedSSD," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 6, p. e6063, 2021.

[4] Q. Gao, J. G. Liu, and Z. J. Ju, "Robust real-time hand detection and localization for space human robot interaction based on deep learning," *Neurocomputing*, vol. 390, pp. 198–206, 2020.

[5] Q. Gao, J. G. Liu, Z. J. Ju, and X. Zhang, "Dual-hand detection for human-robot interaction by a parallel network based on hand detection and body pose estimation," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 12, pp. 9663–9672, 2019.

[6] X. Ma and Z. Zhou, "Student pass rates prediction using optimized support vector machine and decision tree," in *2018 IEEE 8th annual computing and communication workshop and conference (CCWC)*, pp. 209–215, Las Vegas, NV, USA, 2018.

[7] H. Hamsa, S. Indiradevi, and J. Kizhakkethottam, "Student academic performance prediction model using decision tree and fuzzy genetic algorithm," *Procedia Technology*, vol. 25, pp. 326–332, 2016.

[8] M. Ashenafi, G. Riccardi, and M. Ronchetti, "Predicting students' final exam scores from their course activities," in *2015 IEEE Frontiers in education conference (FIE)*, pp. 1–9, El Paso, TX, USA, 2015.

[9] D. Petkovic, M. Sosnick-Pérez, K. Okada et al., "Using the random forest classifier to assess and predict student learning of software engineering teamwork," in *2016 IEEE Frontiers in education conference (FIE)*, pp. 1–7, Erie, PA, USA, 2016.

[10] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: a brief review," *Computational Intelligence and Neuroscience*, vol. 2018, Article ID 7068349, 13 pages, 2018.

[11] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing [review article]," *IEEE Computational Intelligence Magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[12] Y. D. Li, Z. B. Hao, and H. Lei, "Survey of convolutional neural network," *Journal of Computer Applications*, vol. 36, no. 9, pp. 2508–2515, 2016.

[13] P. Kamavisdar, S. Saluja, and S. Agrawal, "A survey on image classification approaches and techniques," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 1, pp. 1005–1009, 2013.

[14] P. N. Druzhkov and V. D. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," *Pattern Recognition and Image Analysis*, vol. 26, no. 1, pp. 9–15, 2016.

[15] M. Ludewig, N. Mauro, S. Latif, and D. Jannach, "Performance comparison of neural and non-neural approaches to session-based recommendation," in *Proceedings of the 13th ACM conference on recommender systems*, pp. 462–466, Copenhagen, Denmark, 2019.

[16] A. Pigeau, O. Aubert, and Y. Prié, *Success Prediction in MOOCs A Case Study*, International Educational Data Mining Society, 2019.

[17] K. Bhumichitr, S. Channarukul, N. Saejiem, R. Jiamthapthaksin, and K. Nongpong, "Recommender systems for university elective course recommendation," in *The 14th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp. 1–5, NakhonSiThammarat, Thailand, 2017.

[18] Z. D. Zhao and M. S. Shang, "User-based collaborative-filtering recommendation algorithms on Hadoop," in *2010 third international conference on knowledge discovery and data mining*, pp. 478–481, Phuket, Thailand, 2010.

[19] Z. A. Pardos, Z. Fan, and W. Jiang, "Connectionist recommendation in the wild: on the utility and scrutability of neural networks for personalized course guidance," *User Modeling and User-Adapted Interaction*, vol. 29, no. 2, pp. 487–525, 2019.

[20] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, pp. 285–295, Minneapolis, MN, 2001.

[21] N. N. Liu, Z. Min, and Y. Qiang, "Probabilistic latent preference analysis for collaborative filtering," in *Proceedings of the 18th ACM conference on Information and knowledge management (CIKM '09)*, pp. 759–766, New York, NY, USA, 2009.

[22] M. Chen, Y. Ma, B. Hu, and L. Zhang, "A ranking-oriented hybrid approach to QoS-aware web service recommendation," in *2015 IEEE International Conference on Services Computing*, pp. 578–585, New York, NY, USA, 2015.

[23] W. Markus, K. Alexandros, Q. V. Le, and S. Alex, "COFI-RANK maximum margin matrix factorization for collaborative ranking," in *Proceedings of the 20th International Conference on Neural Information Processing Systems*, pp. 1593–1600, Red Hook, NY, USA, 2007.

[24] S. Yue, K. Alexandros, B. Linas, L. Martha, and H. Alan, "XCLiMF: optimizing expected reciprocal rank for data with multiple levels of relevance," in *Proceedings of the 7th ACM conference on Recommender systems*, pp. 431–434, New York, NY, USA, 2013.

[25] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-k gains for session-based recommendations," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 843–852, New York, NY, USA, 2018.

[26] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*, pp. 173–182, Republic and Canton of Geneva, Switzerland, 2017.

[27] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[28] B. C. Hardgrave, R. L. Wilson, and K. A. Walstrom, "Predicting graduate student success: a comparison of neural networks and traditional techniques," *Computers & Operations Research*, vol. 21, no. 3, pp. 249–263, 1994.

[29] D. J. Hamilton, *Multiple Regression Analysis and Prediction of GPA upon Degree Completion*, College Student Journal, 1990.

[30] W. R. Klecka, G. R. Iversen, and W. R. Klecka, *Discriminant Analysis*, Sage, 1980.

[31] S. J. Press and S. Wilson, "Choosing between logistic regression and discriminant analysis," *Journal of the American Statistical Association*, vol. 73, no. 364, pp. 699–705, 1978.

[32] T. Wang and A. Mitrovic, "Using neural networks to predict student's performance," in *International Conference on Computers in Education, 2002. Proceedings*, pp. 969–973, Auckland, New Zealand, 2002.

[33] I. Lykourentzou, I. Giannoukos, G. Mpardis, V. Nikolopoulos, and V. Loumos, "Early and dynamic student achievement

prediction in e-learning courses using neural networks," *Journal of the American Society for Information Science and Technology*, vol. 60, no. 2, pp. 372–380, 2009.

[34] H. Sun, C. Yin, H. Chen, L. Qiao, Y. Ouyang, and B. David, "A student's performance prediction method based on neural collaborative filtering," in *2019 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pp. 1–8, Yogyakarta, Indonesia, 2019.

[35] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[36] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. 35, no. 12, pp. 61–70, 1992.

[37] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: an open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pp. 175–186, Chapel Hill, North Carolina, USA, 1994.

[38] M. J. Pazzani and D. Billsus, *Content-Based Recommendation Systems*, The adaptive web. Springer, Berlin, Heidelberg, 2007.

[39] P. C. Chang, C. H. Lin, and M. H. Chen, "A hybrid course recommendation system by integrating collaborative filtering and artificial immune systems," *Algorithms*, vol. 9, no. 3, p. 47, 2016.

[40] R. Li, C. Yin, X. Zhang, and B. David, "Online learning style modeling for course recommendation," in *Recent Developments in Intelligent Computing, Communication and Devices*, Springer, Singapore, 2019.

[41] E. L. Lee, T. Kuo, and S. D. Lin, "A collaborative filtering-based two stage model with item dependency for course recommendation," in *2017 IEEE international conference on data science and advanced analytics (DSAA)*, pp. 496–503, Tokyo, Japan, 2017.

[42] S. Ray and A. Sharma, "A collaborative filtering based approach for recommending elective courses," in *International Conference on Information Intelligence, Systems, Technology and Management*, pp. 330–339, Springer, Berlin, Heidelberg, 2011.

[43] L. Huang, C. D. Wang, H. Y. Chao, J. H. Lai, and P. S. Yu, "A score prediction approach for optional course recommendation via cross-user-domain collaborative filtering," *IEEE Access*, vol. 7, pp. 19550–19563, 2019.

[44] Y. Liu, S. Wang, M. S. Khan et al., "A novel deep hybrid recommender system based on auto-encoder with neural collaborative filtering," *Big Data Mining and Analytics*, vol. 1, no. 3, pp. 211–221, 2018.

[45] Z. Ren, X. Ning, A. S. Lan, and H. Rangwala, "Grade prediction with neural collaborative filtering," in *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10, Washington, DC, USA, 2019.

[46] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," 1998, https://arxiv.org/abs/1301.7363.

[47] G. Kanakana and A. Olanrewaju, "Predicting student performance in engineering education using an artificial neural network at Tshwane University of Technology," in *Proceedings of the ISEM*, Beijing, China, 2011.

[48] M. Gadhavi and C. Patel, "Student final grade prediction based on linear regression," *Indian Journal of Computer Science and Engineering (IJCSE)*, vol. 8, no. 3, pp. 274–279, 2017.