


Article

Predicting Student Achievement Based on Temporal Learning Behavior in MOOCs

Shaojie Qu ¹ , Kan Li ^{2,*}, Bo Wu ³, Shuhui Zhang ³ and Yongchao Wang ⁴

¹ Network Information Technology Center, Beijing Institute of Technology, Beijing 100081, China; qushaojie@bit.edu.cn

² School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China

³ Network Information Technology Center, Beijing Institute of Technology, Beijing 100081, China; wubo@bit.edu.cn (B.W.); zhangshuhui@bit.edu.cn (S.Z.)

⁴ Computer Center, Peking University, Beijing 100871, China; wangyc629@pku.edu.cn

* Correspondence: likan@bit.edu.cn

Received: 12 November 2019; Accepted: 13 December 2019; Published: 16 December 2019



Abstract: With the development of data mining technology, educational data mining (EDM) has gained increasing amounts of attention. Research on massive open online courses (MOOCs) is an important area of EDM. Previous studies found that assignment-related behaviors in MOOCs (such as the completed number of assignments) can affect student achievement. However, these methods cannot fully reflect students' learning processes and affect the accuracy of prediction. In the present paper, we consider the temporal learning behaviors of students to propose a student achievement prediction method for MOOCs. First, a multi-layer long short-term memory (LSTM) neural network is employed to reflect students' learning processes. Second, a discriminative sequential pattern (DSP) mining-based pattern adapter is proposed to obtain the behavior patterns of students and enhance the significance of critical information. Third, a framework is constructed with an attention mechanism that includes data pre-processing, pattern adaptation, and the LSTM neural network to predict student achievement. In the experiments, we collect data from a C programming course from the year 2012 and extract assignment-related features. The experimental results reveal that this method achieves an accuracy rate of 91% and a recall of 94%.

Keywords: smart learning; discriminative sequential pattern; attention mechanism; massive open online course; data science applications in education

1. Introduction

Recent developments in data mining have led to renewed interest in educational data mining (EDM). Studies on massive open online courses (MOOCs) are an important part [1–4] of EDM. Compared to traditional classes, students can obtain optimal learning resources in MOOCs using the internet, without considering the location and cost of obtainment. Researchers are committed to improving the effectiveness of learning in many ways. Previous work has tried to predict and improve student achievement in MOOCs by considering courses [5,6], forums [7], watching behaviors on video [8,9], quizzes [10], plagiarism [11,12], and so on.

Researchers have found that students' behaviors [13] can reflect their learning situation and achievement, and the overall behaviors (such as the number of assignments done, the number of quizzes passed, and total amount of time spent completing assignments [10]) can be used to predict student achievement. Since students' same overall behaviors may have different meanings with different learning process, the use of overall behaviors, which cannot fully reflect students' learning processes, affects the accuracy of predictions. Table 1 shows four students with similar overall behaviors

that have different achievements, and the data used in Table 1 are from the data set mentioned in Section 5.1. All students have completed 69 assignments, and their average submission times of all assignments is 2.42, and their average submission order of all assignments is about 770. However, everyone's score varies greatly from 21 to 80, and this means that the overall behaviors cannot fully reflect students' learning processes.

Table 1. Some overall behaviors of students.

Student ID	Number of Assignments Done	Average Submission Order	Average Submission Times	Score
35366	69	767	2.42	78
34692	69	761	2.42	56
34203	69	786	2.42	21
34677	69	784	2.42	80

The students' learning processes reveal their temporal learning behaviors. Current temporal algorithms that can analyze students' temporal learning behaviors mainly include traditional algorithms [14–17] and deep learning-based algorithms [18–20]. Deep learning-based methods such as long short-term memory (LSTM) have achieved outstanding performance in processing temporal issues. However, as the length of a time-series neural network increases, some critical information will be lost and affect the accuracy of the prediction.

In addition, behavior patterns contain critical behavior information shared by students, and they are also helpful in revealing students' learning processes. Researchers have mined students' behavior patterns [21,22]. Cerezo [21] analyzed log data and revealed four patterns (non-procrastinators, socially focused, individually focused, procrastinators) of online behaviors that predict student achievement. They found that students with procrastination patterns gain low final marks. Kahan [22] identified seven types of participant behaviors. In these behavior patterns, the same behavior pattern may exist in different student groups. For example, the behavior pattern of completing assignments with fewer attempts several times may exist in both the student group with excellent achievements and that with continuous plagiarism. This pattern is not useful for us to predict students' achievements. We need to find discriminative behavior patterns, and these patterns help effectively to predict student achievement.

In order to solve the problems above, in the present paper, we extract temporal assignment-related behaviors (e.g., submission order sequence, completion time sequence) from logs in a course on MOOCs, describe students' learning processes, and predict students' achievement by considering students' learning processes and discriminative behavior patterns.

Our contributions include the following:

- 1 We construct an achievement prediction framework with an attention mechanism that fully leverages student temporal behaviors and behavior patterns.
- 2 We propose a discriminative sequential pattern (DSP) mining algorithm to mine student behavior patterns (such as those defined by Quitter, Faineant, and Cheater), which enhances the importance of critical information and improves the performance of student achievement prediction.
- 3 The experiments show that the accuracy of predicting student achievement using our approach produces a better outcome than some other methods. Our approach obtains an accuracy rate of 91% and a recall of 94%.

The rest of the paper is organized as follows: Section 2 presents analyses of the related work in the field. We present an end-to-end achievement prediction framework in Section 3, and then describe the methods used in Section 4. The experiments and the analyses of the results are shown in Section 5. Section 6 summarizes our work and suggests possibilities for future research.

2. Related Work

2.1. Prediction in MOOCs

Over the past few years, researchers have made substantial progress in mining data from MOOCs, such as mining texts from discussion forums [7], mining video behaviors [8,9], performance prediction [10,13,21,23], plagiarism detection [11,12], and dropout detection [18,19]. These data mined from MOOCs have been widely used for predicting student achievement.

Romero [10] summarized the data mining application for learning with the Moodle system. They used the attributes of the number of assignments done, the number of quizzes done, the number of quizzes passed, the number of messages sent to the teacher, the number of messages sent to the forum, etc., to group the students and predict student performance. Meier et al. [13] proposed a method to increase students learning efficacy in the traditional classroom as well as in the case of MOOCs. They suggested that assessments (such as quizzes) in MOOCs can help to perform timely predictions for each student, allowing the instructor to make appropriate interventions. Cerezo [21] examined the students learning behaviors using data extracted from MOOCs, such as the total time spent on the forum and the number of words posted on the forum, and identified four distinct patterns of the learning process, revealing student groups with different levels of achievement (including the social, individually focused, procrastinator, and non-procrastinator groups). These outcomes have apparent significance for increasing student performance in MOOCs. Conijn [23] reported that 65% of specific course items (e.g., videos started, quizzes finished, resources read) were related to the final exam grade. Students who passed the course spent more time on these items compared to those who failed. However, there was little difference in the sequence of activities among diverse students.

These works mainly consider the overall behaviors of students in MOOCs, which cannot fully reflect students' learning processes, and thus affect prediction accuracy.

2.2. LSTM Networks

The student learning process is a vital consideration for predicting student achievement. For sequential data such as a learning process, temporal algorithms, including traditional algorithms and deep learning-based algorithms, have been widely used. Among these algorithms, recurrent neural network (RNN) [24] or LSTM [25] networks have outperformed other traditional algorithms.

A RNN is a class of artificial neural network, where connections between nodes form a directed graph along a temporal sequence. This allows the network to exhibit a temporally dynamic behavior. Pineda [24] tried to generalize back-propagation to recurrent neural networks, resembling the master/slave network of Lapedes and Farber. However, excessively long input sequences of RNNs will cause gradient explosion or gradient disappearance. LSTM [25,26] networks are designed to fix the shortcomings of a RNN, where it is difficult to process a long sequence. In 2000, Schmidhuber [26] proposed a novel, adaptive forget gate that enables a LSTM cell to learn and reset itself at an appropriate time.

Tang [18] extracted raw features from activity logs and used a RNN with LSTM cells to predict course dropout in MOOCs. Xiong [19] proposed a RNN-LSTM-based prediction model to predict learners' learning status and solve the problem of a high dropout rate. Ding [19] trained a modified auto-encoder combined with a LSTM neural network to select features and predict student performance in MOOCs. The selected features are discriminative for prediction and reduce the overfitting for the low-performing student group. Singh et al. [27] presented a multi-stream, bidirectional RNN for fine-grained action detection. Fei [28] used features such as lecture video watching and forum activities during the study period to predict student dropout.

These above algorithms can deal with the temporal problems well. However, some critical information is lost because of the long length of their time-series neural networks, and thus they affect the accuracy of the prediction.

2.3. Sequential Patterns

Studying different student behavior patterns can also strengthen teaching outcomes [29]. Kahan [22] identified seven types of participant behaviors and termed them tasters (64.8%), downloaders (8.5%), disengagers (11.5%), offline engagers (3.6%), online engagers (7.4%), moderately social engagers (3.7%), and social engagers (0.6%). This study supported the claim that people should evaluate the impact of MOOCs on both certification rates and learning conduct. Rodrigues, Ramos, Silva, and Gomes [30] aimed to discern students’ engagement patterns in MOOC courses using EDM techniques. Their analysis guided the design of adaptive strategies and helped enhance the learning experience. Brinton [31] explored the relationship between students’ video-watching behavior and quiz score in MOOCs. They showed that some behaviors (e.g., skipping back and skipping forward) were significantly correlated with the ability to be ‘correct at first attempt’ (CFA) when answering quiz questions.

For temporal data, sequential patterns can reflect the critical information of data [32]. Sequential patterns were first put forth by Agrawal et al. in 1995 [33]. Other sequential pattern algorithms, such as Aprioriall, AprioriSome, and Dynamicsome [33], were advanced one after another. Subsequently, a new sequential pattern algorithm, the generalized sequential pattern (GSP) [34], was proposed, which is effective at creating sequential patterns, but its efficiency is not high, because the whole sequence must be searched.

Jiang et al. [35] presented a pruning technique and introduced a k-weighted pruning strategy in a weighted, negative sequential pattern mining algorithm. Experiments have shown that their algorithm is successful and produces ideal results.

3. Framework and Data Pre-Processing

In this section, we propose an end-to-end achievement-prediction framework that extracts features from interaction logs in MOOCs and predicts students failing to pass the final exam. As shown in Figure 1, the framework is illustrated in four parts. Part (a) is the data pre-processing, part (b) is a multi-layer LSTM neural network, part (c) is a pattern adapter and part (d) is the attention mechanism.

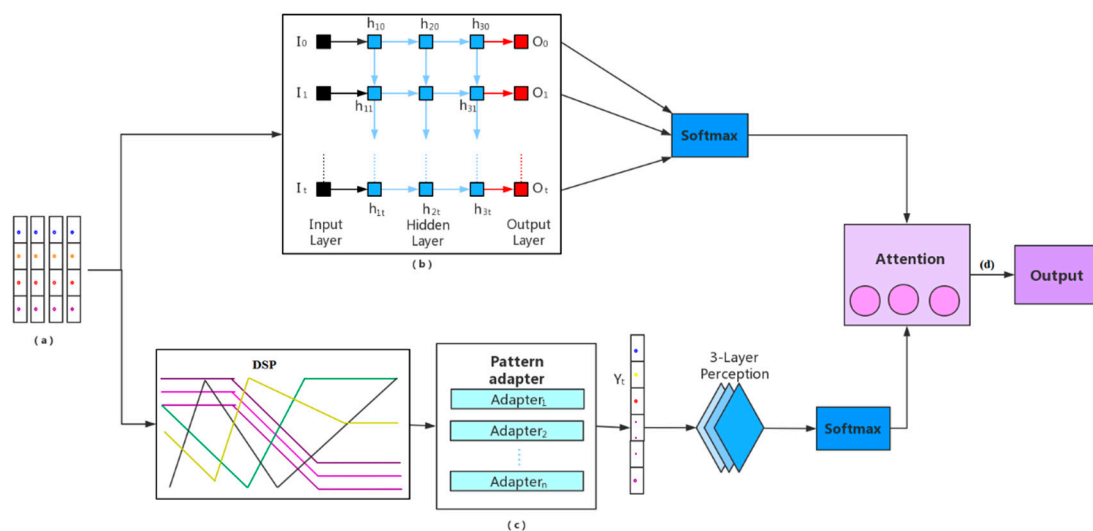


Figure 1. The student achievement-prediction framework in massive open online courses (MOOCs): (a) Data pre-processing; (b) multi-layer long short-term memory (LSTM) neural network for predicting student achievement using temporal behaviors; (c) pattern adapter for mining behavior patterns that could strengthen critical information; (d) attention mechanism to adjust the weight of parts (b) and (c).

We carry out the data pre-processing in part (a) and extract temporal features that reflect the students’ learning process. Part (b) uses a multi-layer LSTM neural network to predict student

achievement, considering students' temporal features. The temporal algorithm-based LSTM neural network, using temporal features, can reflect the learning processes of students well. Part (c) finds the discriminative sequential patterns and constructs pattern adapters according to the sequential patterns. Sequence patterns can emphasize the critical information that may be forgotten by the LSTM neural network in part (c). Part (d) adjusts the weights of parts (b) and (c) by using the attention mechanism, obtaining the final prediction result.

Before extracting temporal features, the data need to be pre-processed. Missing logs or absences of students' assignments will affect the effectiveness of features. Each assignment of each student should have relevant logs, and the logs of each assignment will form a time status with temporal features. Where we found that there were missing logs, these missing logs were replaced with simulated logs. We can do this because all the extracted features are numeric, and thus we can replace the missing data with an adjacent or global value.

There were three approaches used to fill the missing logs: The first approach used was adjacent replenishment, which means to fill the logs by using the average value of adjacent logs. The usage of adjacent replenishment indicates that the learning behavior of a student is similar to that of an upcoming assignment in the process of learning, and the missing behavior data can be filled by the adjacent learning behavior data. The second technique is global replenishment, which involves filling the logs by the average value of all logs of the given student. Global replenishment means that behaviors in the learning processes are similar for the same student, and thus the actual behavior data can be replaced by the average behavior of all assignments. The third method is 0 replenishment, which implies the status value is replaced with 0. This means that the value of all the features reflecting the students' completion of the assignment are set to 0, regardless of whether the students failed to submit their assignment, or if the assignment they submitted was lost because of unexpected circumstances. We provide the experimental outcomes in Section 5.

4. Prediction Method

In this section, we propose an algorithm to predict student achievement, along with the attention mechanism, using the temporal feature sequences. The algorithm can well reflect students' learning processes and can enhance critical information with a DSP-based pattern adapter. By adjusting the weights of the LSTM neural network and DSP-based pattern adapter, better prediction can be obtained.

4.1. Multi-Layer LSTM Neural Networks

We used a multi-layer LSTM neural network to predict students' achievement and reflect students' learning process. The LSTM neural network can remember important information and forget invalid information. This solves the problem of gradient disappearance and gradient explosion in long sequences in RNNs. A LSTM unit consists of an input gate, an output gate, and a forget gate. The input gate inserts the sequential data, and the forget gate selectively forgets the information transmitted by the previous node. The output gate determines the information that can be passed to the next node. The structure of the LSTM cell used is shown in Figure 2.

The calculation process of the output results is shown in Equations (1)–(7). The new memory, C_t , is generated from the output, h_{t-1} and x_t , of the previous unit. Input gate i_t controls the retention of new memory, forgetting gate f_t controls the retention of C_{t-1} before forgetting gate f_t , and the output of final memory is controlled by output gate o_t .

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (1)$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (2)$$

$$\bar{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \quad (3)$$

$$C_t = f_t \circ C_{t-1} + i_t \circ \bar{C}_t \quad (4)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{5}$$

$$C_t = \tanh(C_t) \tag{6}$$

$$h_t = o_t \circ C_t \tag{7}$$

A multi-layer LSTM network generally consists of three layers: (1) An input layer, (2) a hidden layer, and (3) an output layer. Our LSTM adopts a five-layer architecture, including three-layer hidden layers. Part (b) of Figure 1 displays the network.

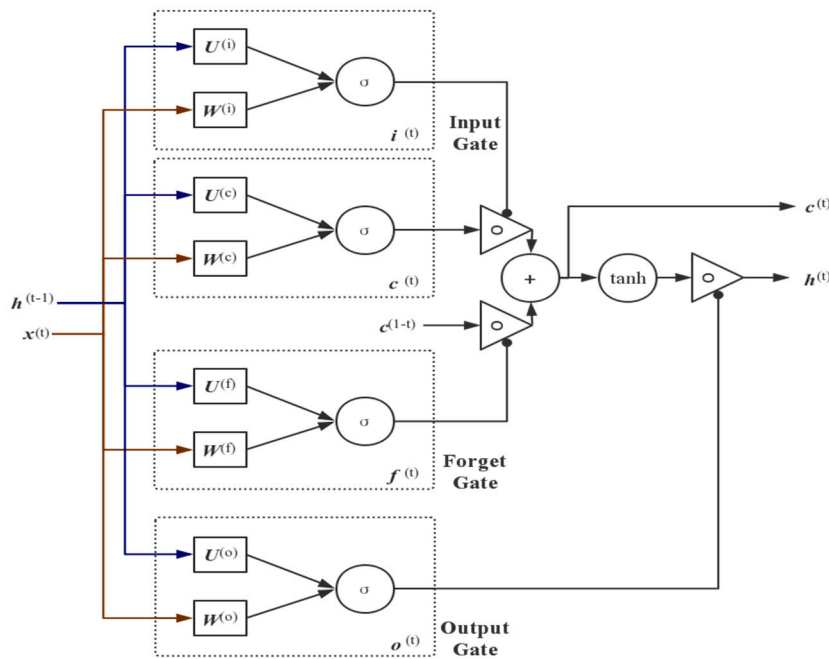


Figure 2. LSTM unit.

4.2. DSP-Based Pattern Adapter

In this part, we use a DSP to mine sequential patterns and construct pattern adapters in order to highlight the importance of critical behavior data. Since LSTM units use the forget gate to forget invalid behavior information, some critical behavior information may also be forgotten. However, sequential patterns contain critical behavior information shared by most sequences and could enhance the importance of information.

Students who pass the exam and the students who fail the exam have different behavior patterns. Taking the students who failed the final exam as an example, we should ensure that the sequential patterns that appear from the students who fail the exam are rarely found in students who pass the exam. Otherwise, the prediction accuracy of sequential pattern prediction will be greatly reduced. The traditional GSP algorithm has shortcomings in dealing with multi-classification sequential pattern prediction. Thus, we propose an improved GSP algorithm, called DSP, to identify students' behavior patterns regarding interval temporal features and construct pattern adapters.

We divided 69 assignments into 14 groups, according to the open time of the assignments. For each student's behavior, we construct a sequence of 14 lengths. The average students' assignment-related behavior in each group was taken as a value of the sequence. We discerned the sequential patterns of the students who pass the final exam (labelled as Classification 0) and those who fail it (labelled as Classification 1).

The DSP algorithm is shown in Algorithm 1. We set two thresholds to control the accuracy and recall of sequential patterns.

Input:

T_{ik} : The set of candidate sequential patterns in the k th iteration of classification i

P_{ik} : The set of frequent sequential patterns of classification i

V : The range of values for the feature sequences

rr and ar : Adjustable parameters to control for the accuracy and recall of obtaining sequential patterns

Output:

CP_i : The classification sequential patterns of classification i

// Generating candidate sequential patterns with a length of 1 according to the range of values in the sequences

$T_{i1} \leftarrow \text{Initialize}()$

//If the recall (RCC _{i}) of candidate sequential pattern f in T_{i1} is greater than rr , then f is added to P_{i1} .

for ($f \in T_{i1}$)

if (RCC _{i} (f) $\geq rr$)

$P_{i1} \leftarrow P_{i1} \cup f$

for ($k = 2; P_{i,k-1} \neq \emptyset; k++$) **do**

// For sequential patterns of length $k-1$ satisfying rr , candidate sequential patterns of length k are generated.

for ($f \in P_{i,k-1}$)

for ($v \in V$)

$T_{ik} = T_{ik} \cup \text{Con}(f, v)$ //The function Con connects the sequence f and value v

endfor

endfor

//If the RCC _{i} of candidate sequential pattern f in T_{ik} is greater than rr , then f is added to P_{ik} .

for ($f \in T_{ik}$)

if (RCC _{i} (f) $\geq rr$)

$P_{ik} \leftarrow P_{ik} \cup f$

//In the sequence satisfying the recall requirement of classification i , if the accuracy of prediction (ACC _{i}) in all classifications is greater than that of ar , it is added to the final result CP_i .

$CP_i = CP_i \cup \{ \langle f \rangle \mid f \in P_{ik}, \text{ACC}_i(f) \geq ar \}$

endfor

return CP_i

We can use the DSP algorithm to obtain many behavior patterns, which constitute the corresponding pattern adapters. For sequential patterns, we can construct a n -dimensional vector, where there are n patterns. A feature sequence will match these sequential patterns separately. If the matching is successful, the corresponding position of the vector is 1. If the matching fails, the corresponding position of the vector is 0. A student's behaviors can be transformed into the n -dimensional vector through the pattern adapters, which can be trained by a simple neural network to predict student achievement.

4.3. Attention Mechanism

We used the attention-based student achievement prediction model to evaluate the importance of the LSTM neural network and sequence patterns adapter. The attention mechanism originates from the way of thinking of the human brain [36], which automatically pays more attention to vital details and ignores less important information.

Note that each output of the multi-layer LSTM neural network, such as that from O_1 to O_t in part b , is not normalized, so we need to score each output, as shown in the following equation:

$$S_{ib} = \text{softmax}(O_i) \quad i \in (1, 2, \dots, t) \quad (8)$$

Similarly, we normalize output O_c of part c . The formula is displayed in Equation (11).

$$S_c = \text{softmax}(O_c) \quad (9)$$

The final score in the prediction process is composed of parts b and c . We use the attention mechanism to adjust the weights of parts b and c . The weight of part b for output O_i is W_i , and that of part c is W_c . We used the feed forward neural network to train the model. The network was trained jointly with the multi-layer LSTM neural network in part b and the multi-layer perceptron in part c , shown in Figure 1. The final loss function (LOSS) is composed of the loss of parts b and c , where L is the softmax loss function. The LOSS function is presented as follows.

$$\text{LOSS} = L(S_c) * W_c + W_b \left(\sum_{i=0}^t L(S_{ib}) * V_i \right) \quad (10)$$

$$\sum_{i=0}^t V_i = 1 \quad (11)$$

$$W_c + W_b = 1 \quad (12)$$

We performed back propagation to train the parameters via a gradient descent algorithm.

5. Experiments

5.1. Data and Distribution

In this section, we introduce the data set and describe the features extracted from the logs. Finally, the data distribution is presented.

The data are from a course called 'C Programming', from 2012. In total, 1528 students took part in this course. The course includes 69 programming assignments. Students are supposed to complete these assignments, and thirty-four features can be extracted from the logs as shown in Table A1. We have extracted the features of the assignment behaviors and some compiled error information. These features (including assignment submission order, quick submission times, runtime error (RE) times, completion time, and compile information, etc.) denote the students' completion of assignments from multiple angles, and thus reflect the students' learning status.

Figure 3 is a programming assignment to be completed by students. As there are many difficult hidden test cases, it is difficult for students to pass all the test cases at once.

The assignment contains five test cases. The first three are public, and the last two are hidden test cases that students cannot see. After the student submits the program, the platform compiles the student's program, enters the test case, judges whether the output is consistent with the expected output, and if it is consistent, judges that the student has passed the test case.

A submission order refers to the order of submitting an assignment to the website. For a student, if he/she is the first to submit the assignment, the submission order for the assignment is 1. If he/she is the second student, the submission order for the assignment is 2. Among these programming assignments, students will submit the assignment many times to pass all test cases in order to obtain a high score.

A quick submission is determined to occur when the submission time minus the browsing time is less than 300 s. The browsing time begins measurement when a student first reads the assignment, while the submitting time is measured when the student submits it. Since students need to view the test cases online for specific output formats and adjust the output format of the program by referring to the test cases, students are less likely to view the assignment elsewhere and remember the output format to complete the assignment. There is a large probability that a quick submission means the student copied the assignment from someone else. The following figures demonstrate the correlation between average quick submission times and academic performance.

Questions:

Programming to achieve Number System Conversion

Input:

The original number system, the target number system and the number to be converted.

Output:

The value of number in the target number system.

Test Case

No.	Public	Test Input	Test Output	Time limitation	Weight
1	Always	10, 2, 12	1100	1 seconds	1
2	Always	2, 8, 3	error value	1 seconds	1
3	Always	8, 16, 456	12E	1 seconds	1
4	Never	2, 16, 111111111111	FFF	1 seconds	1
5	Never	16, 2, H	error value	1 seconds	1

Figure 3. A programming assignment for number system conversion.

Figure 4 reveals the relationship between the score in the final exam and the average quick submission times in different scores. It shows that the better the score, the lower the average quick submission times. Although we cannot say that quick submission must indicate plagiarism, there is a large probability that quick submission has something to do with plagiarism. Students with scores below 10 have fewer quick submission times, as they do not complete enough assignments.

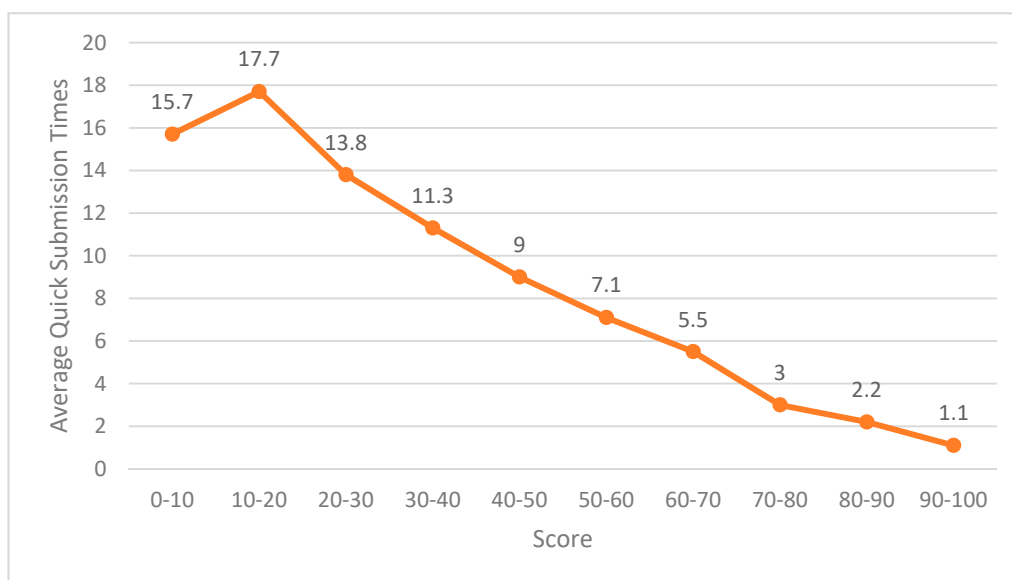


Figure 4. The relationship between the score in the final exam and average quick submission times.

Table 2 reveals the relationship between submission orders and exam scores. For students with scores under 10, their average submission order of all assignments is 1061, while for students with scores above 90, their average submission order of all assignments is 393. This finding indicates that students who submit their assignments early have higher learning enthusiasm and students with higher learning enthusiasm are likely to receive higher scores.

Table 2. The relationship between average submission order and exam scores.

Exam Scores of Students	Average Submission Order of All Assignments
0–10	1061
10–20	1002
20–30	949
30–40	884
40–50	832
50–60	759
60–70	710
70–80	612
80–90	555
90–100	393

Figure 5 reveals the relationship between the scores of students and their submissions times during the learning process. Students with a score between 50 and 60 try to submit their assignments more times in the early 10 assignments, and, in the later phases, students with a score between 90 and 100 try to submit their assignments more times. This indicates that some students with a poor score try to complete their assignments with more attempts in the early part, showing evidence of their poor learning ability. However, in the later part, they often submit assignments once and pass all test cases, as they copy the assignments from other students and have fewer submission times. Students with a score between 90 and 100 can complete easy assignments with fewer attempts in the early stage. After the assignments become more complex, they are willing to try more to complete the assignments independently.

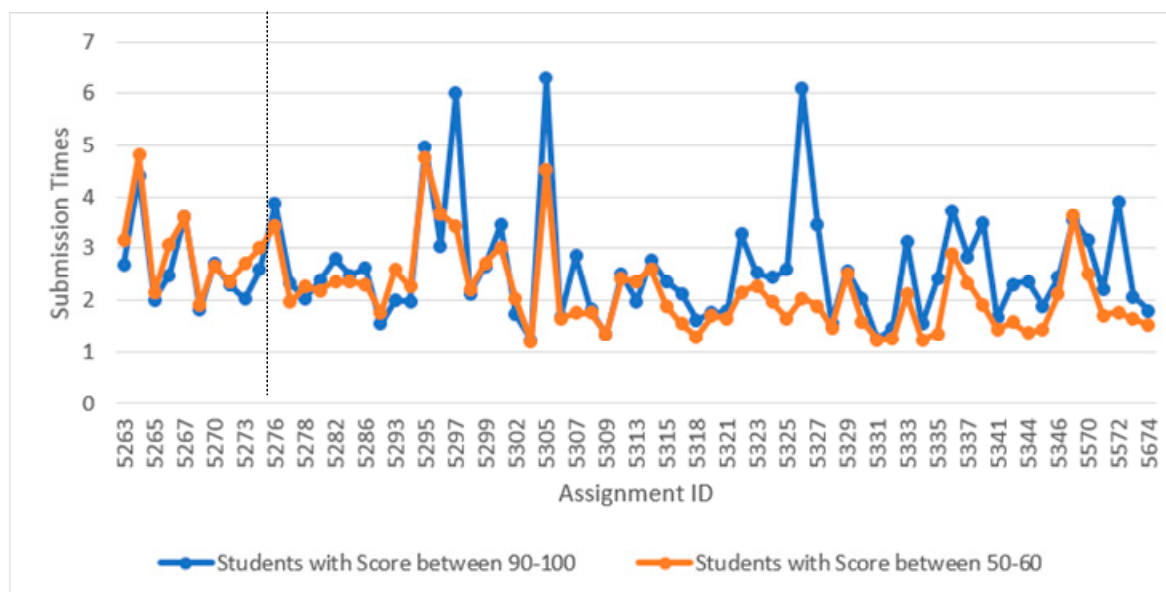


Figure 5. The relationship between submission times and score.

5.2. Baselines

In the experiment, we test the standard multi-layer perceptron (MLP), standard LSTM, M-F-LSTM [37], NOSEP [38], and our method.

A LSTM-based neural network is appropriate for sequence data. We used LSTM to make two experiments according to different assignment groups. This course lasts for 14 weeks and has 69 assignments. In the first experiment, we extracted the corresponding features from each assignment in the unit of assignment and set the unit number of LSTM to 69. In the second experiment, we

extracted the corresponding features from all the assignments in each week and set the unit number of LSTM to 14.

M-F-LSTM is a multi-layer LSTM that it is fully connected among layers.

NOSEP is a sequence pattern mining method that can be used to find the students' behavior patterns. We can use the behavior patterns to predict student performance.

Our method fully considers the learning process of students and pays attention to critical information.

5.3. Evaluation Metrics

In this paper, we evaluate the performance of the data pre-processing method, the sequential pattern mining method, and the students' overall performance prediction method, respectively.

We use the accuracy and recall rates to compare the performance differences of each algorithm in the data pre-processing method.

Here, we define two evaluation metrics to evaluate the DSP algorithm: If each value of sequence pattern f appears in sequence s in turn, we define that sequence s is in accordance with sequence pattern f . For example, $s_1 = (a, d, c, c, b, b, e, e, e, e, e, e, e)$ and $s_2 = (a, b, c, d, b, d, e, e, e, e, e, e, e)$ are two students' behavior sequences. For sequential patterns $f_1 = (a, c, d)$ and $f_2 = (a, c, b)$, a, c , and d in f_1 appear in sequence s_2 , so s_2 is in accordance with f_1 . Similarly, s_1 and s_2 are in accordance with f_2 .

Definition 1. The recall of sequential pattern f in classification i (RCC_i) equals M_{ci} divided by N_i , where M_{ci} is the number of students whose behavior sequences are in accordance with pattern f in classification i . N_i is the overall number of students of classification i . RCC_i refers to the probability that the student behaviors sequences could in accordance with sequential pattern f in classification i . The formula for this is as follows:

$$RCC_i = M_{ci}/N_i \quad (13)$$

Definition 2. The accuracy of sequential pattern f (ACC_i) in classification i equals M_c divided by N , where M_c is the number of students whose behavior sequences are in accordance with pattern f in all classifications. N is the number of students in all classifications. ACC_i refers to the proportion of the true result when predicting with sequential pattern f . The formula for this is as follows:

$$ACC_i = M_c/N \quad (14)$$

We used accuracy and recall to compare the performance differences of each algorithm and used 5-fold cross validation to verify the stability of the algorithms in the students overall performance prediction.

5.4. Training Details

We predicted whether the students' performance in the final exam is greater than 70 points by examining the students' behavior. We think that students with more than 70 points have better performance, and students with less than 70 points have difficulties in learning. We trained neural networks with 80% of the data and validated with the remaining 20% of data.

We use the Tensor Flow framework to train the data using the GPU. For the LSTM neural network in our method, we used three hidden layers, each with 69 units. We set 'learning_rate' to 0.0001, 'batch_size' to 64, 'n_hidden' to 64, and conducted 500,000 iterations of training. In the DSP algorithm for finding students' behavior patterns, we set ACC to 0.7 and RCC to 0.7, in order to take into account both the accuracy and recall of the model.

5.5. Results

5.5.1. Data Pre-Processing

For existing assignment-related behaviors in MOOC logs, we can quickly extract relevant features and construct behavior sequences. However, for uncommitted or unexpectedly missing assignments, we need to use algorithms to repair these behaviors and build behavior sequences.

Each of the 1528 students needed to complete 69 assignments. When we use the LSTM network for prediction, the number of assignments in the logs should be 105,432 (1528 × 69), however, the actual number of assignments was only 104,194, which equates to a missing feature rate of 1.2%. We tested three methods to simulate the missing logs, as described earlier.

The outcome in Table 3 reveals that the 0-replenished method leads to the best performance. In other words, missing features (substituting 0) better reflect the learning state of students who fail to submit assignments, while global or adjacent feature substitution will make the predicted results less accurate. The performance of the 0-replenished method shows that a value of zero can better reflect the learning state of students and implicitly shows that a student has not submitted their assignment.

Table 3. The comparison of three methods that fill missing logs.

Method	Accuracy	Recall
Adjacent-replenished	89%	94%
0-replenished	90%	94%
Global-replenished	87%	89%

5.5.2. DSP-Based Pattern Adapter

We constructed feature sequences of 14 lengths per week for multiple features. In order to find behavioral patterns, we need to discretize the values of features. Take the submission order as an example: Since there are 1528 students, the order of submitting an assignment is 1–1528. The first submission is 1, and the last submission is 1528. We discretized the value of the submission order, and the eigenvalue of the first 500 submitting students was set to be value ‘1’. The eigenvalues of the 501–1000th students who submitted assignments were discretized to value ‘2’. The eigenvalues of the 1001–1528th students who submitted assignments were discretized to value ‘3’. Similarly, we discretized other features. Then, we used the DSP algorithm to find behavior patterns. The experimental results are as follows.

The sequential patterns selected in the experiment are all behavioral patterns where students fail the final exam. For students who pass the exam, it is difficult to find sequential patterns, as losers have the same reasons, while winners have their own strategies for success. Several sequential patterns are selected in Table 4. In Table 4, (·) means any number of values, ranging from 0 to infinity.

Table 4. Behavior patterns.

Pattern Name	Feature	Pattern	Accuracy	Recall	Pattern Type
Quitter 1	Submission times	(·) 1 (·) 2 (·) 2 (·)-2 (·)-2 (·)	72.7%	72.1%	Failed exam
Quitter 2	Submission times	(·) 2 (·) 2 (·) 2 (·)-2 (·)-2 (·)	73.6%	73.9%	Failed exam
Faineant	Submission order	(·) 3 (·) 3 (·) 3 (·)	73.8%	70.2%	Failed exam
Cheater	Plagiarism	(·) 2 (·)	73.49%	85.00%	Failed exam

The first two sequential patterns, called Quitter 1 and Quitter 2, respectively, reflect the learning behavior of students with learning difficulties. In these two patterns, a value of ‘2’ means that the number of submissions is much larger than the average number of submissions of all people in the group, and a value of ‘-2’ denotes that the number of submissions is much smaller than the average

number of submissions. These students have trouble learning in the early stages and need to spend more time completing their assignments. However, in the late phases of learning, they give up doing assignments independently and complete them through plagiarism, so their submission times are far less than the average submission times.

The third sequential pattern, called *Faineant*, means that students do not study actively and always submit their assignments at the end. In the submission order, value '3' means that the student submitted his assignment after 1000 students. The experimental outcomes demonstrate that if three groups of assignments submission order are after 1000, the probability of failure of these students is 73.8%. This indicates that the order of submission can reflect students' learning initiative and has an important impact on the final exam.

The fourth sequential pattern, called *Cheater*, means that students have serious cheating in their assignments. This means the student has committed serious fraud for a group of assignments. The experimental findings show that 73.49% of the students who cheat seriously for a group of assignments fail the final exam.

5.5.3. Prediction Results

The simulated result of our method is shown in Figure 6. After 500,000 iterations, the loss gradually converges, and the accuracy approaches 100% in the training data set due to possible overfitting. The final outcome achieved is an accuracy rate of 91% in the testing data set.

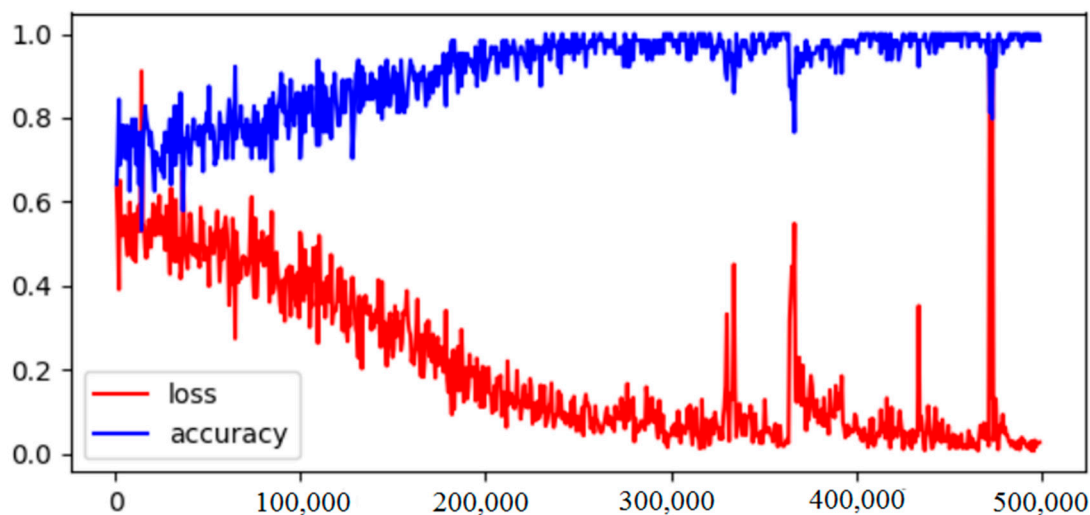


Figure 6. The iteration process of our method.

As shown in Table 5, the MLP method with global features achieves an accuracy of 75%. In contrast, the LSTM ($l = 14$) method achieves an accuracy of 80%, while the other LSTM ($l = 69$) method achieves an accuracy of 88%, and the M-F-LSTM method achieves an accuracy of 91%. The results reveal that temporal algorithms (such as LSTM and M-F-LSTM) can better reflect the learning processes of students and achieve better prediction result. The variable value of the global feature in MLP method may be the same for different behaviors. For example, two students completed the first 30 assignments and the last 30 assignments in 69 assignments, respectively, and the values of the assignment completion number in overall features were both 30 for both students. However, the same values reflect different behaviors. A student who has completed the first 30 questions gave up learning later, while a student who has completed the last 30 questions began to work hard later. As there is a deadline for an assignment, the student who has completed the last 30 assignments cannot make up for the previous assignments. The results indicate that the MLP method with global features cannot distinguish the learning state of these two students, while temporal methods can well reflect the learning process of these two students and

distinguish the learning state of these two students. The LSTM ($l = 69$) performance was better than the other LSTM ($l = 14$) method, as more units can better reflect students' learning processes.

Table 5. Comparisons among our method and other approaches. MLP: Multi-layer perceptron. LSTM: Long short-term memory. NOSEP: Nonoverlapping Sequence Pattern Mining. M-F-LSTM: Multi-task Fully-connected LSTM.

Simulation Method	Accuracy	Recall	Cross Validation
MLP	75%	75%	0.75
LSTM ($l = 14$)	80%	92%	0.85
LSTM ($l = 69$)	88%	94%	0.91
NOSEP	86%	85%	0.85
M-F-LSTM	91%	92%	0.92
Our Method	91%	94%	0.93

NOSEP, a sequence pattern algorithm, can find student behavior patterns and predict students' performance. This method pays more attention to students' critical information. The NOSEP method achieved the accuracy of 86% and a recall of 85%.

Although temporal algorithms can better reflect the learning process of students, some information may be forgotten with the LSTM method. We synthesized the LSTM method and behavior pattern method to reflect the learning process and enhance the importance of critical information. Our method achieves an accuracy of 91% and a recall of 94%, which performs better than the other methods.

6. Conclusions

Previous studies have found that assignment-related behavior in MOOCS can influence student achievement. However, previous studies have not fully considered the pertinent learning process. In this paper, we aim to explore the influence of the learning process on student achievement. A framework was constructed with an attention mechanism that includes data pre-processing, a DSP-based adapter, and a LSTM-based neural network to predict student achievement. We used the LSTM neural network to reflect the students' learning processes and the DSP-based adapter to enhance the significance of critical information. Some sequence patterns were mined such as the Quitter, Faineant and Cheater patterns. The experimental results also show that students who failed in the exam had common sequence patterns, while those who passed the exam did not have common sequence patterns. This means that all roads lead to Rome, but failures are always the same. Our method achieves an accuracy rate of 91% and a recall rate of 94% when predicting student achievement. Thus, our study makes an important contribution to the literature on finding the relationship between the learning process and student achievement.

In future work, we will examine the coding content of programming assignments for higher student performance and will consider the universality of the research on MOOC courses.

Author Contributions: Conceptualization, S.Q. and Y.W.; Methodology, S.Q.; Software, B.W. and S.Z.; Validation, B.W.; Formal analysis, S.Q.; Investigation, S.Q.; Resources, S.Q.; Data curation, Y.W.; Writing—original draft preparation, S.Q.; Writing—review and editing, K.L.; Visualization, S.Q.; Supervision, S.Q.; Project administration, S.Q.; Funding acquisition, K.L.

Funding: This research was funded by Key Program of the National Natural Science Foundation of China, grant number 71834001.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Appendix A

Table A1. Some of the features.

Feature	Meaning
Final Submission Order	Order of submitting one assignment to the website for the final time
Quick Submission Times	Quick submission times of all assignments
Submission Times	Value of submission times for assignments
PE Error Times	Presentation error (PE) times in compiled information
RE Error Times	Runtime error (RE) times in compiled information
TLE Error Times	Time limit exceeded (TLE) times
WA Error Times	Wrong answer (WA) times
Pass Number	Total number of assignments for which answers are correct
Pass Rate	Rate at which assignment answers are correct in all assignments
First Submission Order	Order of submitting one assignment to the website for the first-time
Completion Duration	Completion duration for assignments
Submission Status	Submission status for assignments
One-Time Pass Number	Number passing all cases for one submission in all assignments
One-Time Pass Rate	Rate at which assignment answers pass all test cases in all assignments submitted once
Programming Code Lines	value of programming code lines
Completion Time	The time when a student completed an assignment
Time View Assignment	The time when a student viewed an assignment
Time Submit Assignment	The time when a student submitted an assignment
Test Case Pass Number	The number that how many test cases were passed in on submission
Sum Time Used	Sum consumed time by program running
Average Memory Used	Average consumed memory by program running

References

- Pang, Y.; Song, M.; J, Y.; Zhang, Y. Survey of MOOC related research. In Proceedings of the International Conference on Database Systems for Advanced Applications, Hanoi, Vietnam, 20–23 April 2015. [CrossRef]
- Zhang, H.; Huang, T.; Lv, Z.; Liu, S.Y.; Zhou, Z. MCRS: A course recommendation system for MOOCs. *Multimed. Tools Appl.* **2018a**, *77*, 7051–7069. [CrossRef]
- Rodríguez, B.C.P.; Bird, T.; Conole, G. Evaluation of massive open online courses (MOOCs): A case study. Global Learn, Association for the Advancement of Computing in Education (AACE): Berlin, Germany, April 2015. Available online: <https://www.learntechlib.org/primary/p/150900/> (accessed on 11 November 2019).
- Romero, C.; Ventura, S. Educational data science in massive open online courses. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2016**, *7*, e1187. [CrossRef]
- Baher, S.; Lobo, L.M.R.J. Best combination of machine learning algorithms for course recommendation system in e-learning. *Int. J. Comput. Appl.* **2013**, *41*, 1–10. [CrossRef]
- Yu, F.; Zheng, D. Education data mining: How to mine interactive text in MOOCs using natural language process. In Proceedings of the 12th International Conference on Computer Science and Education (ICCSE), Houston, TX, USA, 22–25 August 2017. [CrossRef]
- Almatrafi, O.; Johri, A.; Rangwala, H. Needle in a haystack: Identifying learner posts that require urgent response in MOOC discussion forums. *Comput. Educ.* **2018**, *118*, 1–9. [CrossRef]
- Chiu, Y.C.; Hsu, H.J.; Wu, J.; Yang, D.L. Predicting student performance in MOOCs using learning activity data. *J. Inf. Sci. Eng.* **2018**, *34*, 1223–1235. [CrossRef]
- Liang, J.; Yang, J.; Wu, Y.; Li, C.; Zheng, L. Big data application in education: Dropout prediction in Edx MOOCs. In Proceedings of the 2016 IEEE Second International Conference on Multimedia Big Data (BigMM), Taipei, Taiwan, 20–22 April 2016. [CrossRef]
- Romero, C.; Ventura, S.; García, E. Data mining in course management systems: Moodle case study and tutorial. *Comput. Educ.* **2008**, *51*, 368–384. Available online: http://www.researchgate.net/publication/222675404_Data_ (accessed on 10 October 2019). [CrossRef]

11. Akçapınar, G. How automated feedback through text mining changes plagiaristic behavior in online assignments. *Comput. Educ.* **2015**, *87*, 123–130. Available online: <http://dl.acm.org/citation.cfm?id=2838222>. (accessed on 10 October 2019). [[CrossRef](#)]
12. Northcutt, C.G.; Ho, A.D.; Chuang, I.L. Detecting and preventing ‘multiple-account’ cheating in massive open online courses. *Comput. Educ.* **2016**, *100*, 71–80. [[CrossRef](#)]
13. Meier, Y.; Xu, J.; Atan, O.; Van Der Schaar, M. Personalized grade prediction: A data mining approach. In Proceedings of the 2015 IEEE International Conference on Data Mining, Atlantic City, NJ, USA, 14–17 November 2015. [[CrossRef](#)]
14. Yang, T.Y.; Brinton, C.G.; Joe-Wong, C.; Chiang, M. Behavior-Based Grade Prediction for MOOCs via Time Series Neural Networks. *IEEE J. Sel. Top. Signal Process.* **2017**, 716–728. [[CrossRef](#)]
15. Faucon, L.; Kidziski, L.; Dillenbourg, P. Semi-Markov model for simulating MOOC students. In Proceedings of the 9th International Conference on Educational Data Mining, EDM, Raleigh, NC, USA, 29 June–2 July 2016; pp. 358–363.
16. Zhang, L. Evaluation of Teaching Effectiveness Based on Gray Markov Chain in the Context of MOOC. In Proceedings of the 2019 2nd International Conference on Advanced Materials, Intelligent Manufacturing and Automation, Zhuhai, China, 17–19 May 2019; p. 052042. [[CrossRef](#)]
17. Baier, C.; Haverkort, B.; Hermanns, H.; Katoen, J.P. Model-checking algorithms for continuous-time Markov chains. *IEEE Trans. Softw. Eng.* **2003**, *29*, 524–541. [[CrossRef](#)]
18. Cui, T.; Yuanxin, O.; Wenge, R.; Jingshuai, Z.; Zhang, X. Time series model for predicting dropout in massive open online courses. In Proceedings of the Artificial Intelligence in Education. 19th International Conference, AIED 2018, Cham, Switzerland, 27–30 June 2018; pp. 353–357. [[CrossRef](#)]
19. Xiong, F.; Zou, K.; Liu, Z.; Wang, H. Predicting learning status in MOOCs using LSTM. In Proceedings of the 2019 ACM Turing Celebration Conference-China, ACM TURC 2019, Chengdu, China, 17–19 May 2019. [[CrossRef](#)]
20. Ding, M.; Yeung, D.Y.; Yang, K.; Pong, T.C. Effective feature learning with unsupervised learning for improving the predictive models in massive open online courses. In Proceedings of the 9th International Conference on Learning Analytics and Knowledge, LAK 2019, Tempe, AZ, USA, 4–8 March 2019; pp. 135–144. [[CrossRef](#)]
21. Cerezo, R.; Sánchez-Santillán, M.; Paule-Ruiz, M.P.; Núñez, J.C. Students’ LMS interaction patterns and their relationship with achievement: A case study in higher education. *Comput. Educ.* **2016**, *96*, 42–54. [[CrossRef](#)]
22. Kahan, T.; Soffer, T.; Nachmias, R. Types of participant behavior in a massive open online course. *Int. Rev. Res. Open Distrib. Learn. (IRRODL)* **2017**, *18*. [[CrossRef](#)]
23. Conijn, R.; Van den Beemt, A.; Cuijpers, P. Predicting student performance in a blended MOOC. *J. Comput. Assist. Learn.* **2018**, *34*, 615–628. [[CrossRef](#)]
24. Pineda, F.J. Generalization of back-propagation to recurrent neural networks. *Phys. Rev. Lett.* **1987**, *59*, 2229. [[CrossRef](#)] [[PubMed](#)]
25. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)] [[PubMed](#)]
26. Gers, F.A.; Schmidhuber, J.; Cummins, F. Learning to forget: Continual prediction with LSTM. *Neural Comput.* **2000**, *12*, 2451–2471. [[CrossRef](#)]
27. Singh, B.; Marks, T.K.; Jones, M.; Tuzel, O.; Shao, M. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016. [[CrossRef](#)]
28. Fei, M.; Yeung, D.Y. Temporal Models for Predicting Student Dropout in Massive Open Online Courses. In Proceedings of the 15th IEEE International Conference on Data Mining Workshop, ICDMW 2015, Atlantic City, NJ, USA, 14–17 November 2015; pp. 256–263. [[CrossRef](#)]
29. Qu, S.; Li, K.; Zhang, S.; Wang, Y. Predicting Achievement of Students in Smart Campus. *IEEE Access* **2018**, *6*, 60264–60273. [[CrossRef](#)]
30. Rodrigues, R.L.; Ramos, J.L.C.; Silva, J.C.S.; Gomes, A.S. Discovery engagement patterns MOOCs through cluster analysis. *IEEE Latin Am. Trans.* **2016**, *14*, 4129–4135. [[CrossRef](#)]
31. Brinton, C.G.; Buccapatnam, S.; Chiang, M.; Poor, H.V. Mining MOOC clickstreams: Video-watching behavior vs. in-video quiz performance. *IEEE Trans. Signal Process.* **2016**, *64*, 3677–3692. [[CrossRef](#)]

32. Neto, H.C.; Julia, R.M.S. ACE-RL-checkers: Decision-making adaptability through integration of automatic case elicitation, reinforcement learning, and sequential pattern mining. *Knowl. Inf. Syst.* **2018**, *57*, 603–634. [[CrossRef](#)]
33. Agrawal, R.; Srikant, R. Mining sequential patterns. In Proceedings of the Eleventh International Conference on Data Engineering, Taipei, Taiwan, 6–10 March 1995. [[CrossRef](#)]
34. Srikant, R.; Agrawal, R. Mining sequential patterns: Generalizations and performance improvements. In Proceedings of the International Conference on Extending Database Technology, Avignon, France, 25–29 March 1996. [[CrossRef](#)]
35. Jiang, H.; Ning, X.; Xie, Q. Research on pruning techniques of mining weighted sequential patterns. In Proceedings of the 2018 International Conference on Internet and e-Business, Singapore, 25–27 April 2018. [[CrossRef](#)]
36. Koch, C.; Ullman, S. Shifts in Selective Visual Attention: Towards the Underlying Neural Circuitry. *Hum. Neurobiol.* **1987**, *4*, 219–227. [[CrossRef](#)]
37. Wang, C.; Yang, H.; Meinel, C. Image captioning with deep bidirectional LSTMs and multi-task learning. *ACM Trans. Multimed. Comput. Commun. Appl.* **2018**, *14*, 1–20. [[CrossRef](#)]
38. Wu, Y.X.; Tong, Y.; Zhu, X.Q.; Wu, X.D. NOSEP: Nonoverlapping Sequence Pattern Mining with Gap Constraints. *IEEE T. Cybern.* **2018**, *48*, 2809–2822. [[CrossRef](#)] [[PubMed](#)]



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).