Computers and Electrical Engineering 93 (2021) 107271

Contents lists available at ScienceDirect

Computers and Electrical Engineering

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

Deep analytic model for student dropout prediction in massive open online courses

Ahmed A. Mubarak [a,b], Han Cao [a], Ibrahim M. Hezam [c,*]

[a] School of Computer and Science- Shaanxi Normal University,-Xian 710119, China
[b] Department of Computer, Ibb University, Ibb, Yemen
[c] Statistics & Operations Research Department, College of Sciences, King Saud University, Riyadh, Saudi Arabia

ARTICLE INFO

Editor: Dr. M. Malek

Keywords:
Convolutional neural networks (CNNs)
Long short-term memory (LSTM)
Class imbalance
Cost-sensitive
Dropout prediction
Feature extraction
Massive Open Online Courses (MOOCs)

ABSTRACT

Predicting students' performance is critical in Massive Open Online Courses (MOOCs) in order to benefit from many aspects such as students' retention and make timely interventions. In this paper, we propose a hyper-model of Convolutional Neural Networks and Long Short-Term Memory, called CONV-LSTM, in order to automatically extract features from MOOCs raw data and predict whether each student will drop out or complete courses. We consider class imbalance problem, which means that models will be biased to yield good results on the majority of class examples and poor results on the minority of class examples. In that case, model prediction is inaccurate, which means that the false negative rate is high. To reinforce better prediction performance, a cost-sensitive technique is used in the loss function, which considers the various misclassification costs for false negatives and false positives. The proposed model shows a better performance when compared to baseline methods.

1. Introduction

As the coronavirus (COVID-19) pandemic of 2020 intensified, people were forced to stay at home to limit the outbreak of the virus. Thus, in order to continue pace with the development of technology and online education, most universities headed to leverage online learning platforms to continue with the educational process. Online learning platforms provide courses in form of video lectures, discussion forums, online assessments, and even live video discussions. This convenient learning method adopts the methodology of crossing the boundaries of time and space in order to enable a significant number of students to enroll at anytime and anywhere in such courses. Having been different from traditional education, a direct contact exists between students and instructors in a specific place and time (classroom). Recently, online learning has promoted a new form of educational process development, attracting a large number of stakeholders and students, where students basically interact directly with systems/applications instead of instructors. Nevertheless, there are still some challenges in how to assess students' performance comprehensively from different perspectives.

Learning analytics which introduced as a useful method to analyze online learning is usually done by analyzing students' profiles such as demographic data and students' interaction records like clickstream behavior, which is stored in the platform. Therefore, many predictive methods e.g., Chui et al. [1]., Waheed et al. [2] and Xing & Du [3] tried to predict the outcome of each student's performance in a course such as failure, passing, and prediction of his/her status (continuous or dropping out). Such methods ultimately help

This paper is for special section VSI-tei. Reviews processed and recommended for publication by Guest Editor Dr. Samira Hosseini.

* Corresponding author.
E-mail addresses: ahmedmubarak@ibbuniv.edu.ye (A.A. Mubarak), caohan@snnu.edu.cn (H. Cao), ialmishnanah@ksu.edu.sa (I.M. Hezam).

https://doi.org/10.1016/j.compeleceng.2021.107271
Received 25 September 2020; Received in revised form 4 April 2021; Accepted 14 June 2021
Available online 30 June 2021

instructors to observe students' performance and timely intervene in order to limit dropout and failure rates Nevertheless, dropout rates in online learning are still high, which is regarded as one of the prominent problems.

Most of the predictive methods depend on two basic steps. The first step is to use feature engineering to extract features from different students' behavior logs. The second step is that the extracted features are analyzed, and then student's state is predicted using predictive methods such as machine learning techniques [4] and deep learning [3]. Although these methods have achieved desirable results, some problems remain unsolved. One significant problem is the features extraction method in the first step, in most of the proposed methods, the feature of extraction is achieved by feature engineering [5].

Feature Engineering is a technique of extracting features manually made by people who are familiar with raw dataset structure for online learning students' logs and have some in-depth knowledge of this field. Usually, it's termed as handcrafted feature engineering. Extracting features requires many iterations processes of extracting and testing, and it takes very long time. So, feature engineering is limited by both human time restrictions and vision. A question is raised here which is "How can we conceptualize every likely valuable feature?". This leads to bias for some features regardless the fact that it may add value to the prediction model or may not. Besides, effective feature engineering strategies for a particular dataset may not be effective for a different dataset, because each Online Learning system stores students' logs with different characteristics of format and content. Plus, new platforms may emerge and produce new datasets that have different characteristics for new courses. Therefore, these hindrances in feature engineering represent particularly serious problems in learning analytics in MOOCs. In this case, there is a requirement to evolve new methods that can avoid the problems of feature engineering. For instance, deep learning is a model that has been smoothly employed in multi fields such as Natural Language Processing, image recognition and learning analytics [6]. A notable feature of deep learning approaches is that they can extract significant features automatically from raw data [6]. A Convolutional neural network (CNN) is one of the advantages of deep learning to automatically extract features. It does not require much manual manipulation of feature extraction[3]. Besides, CNN captures all potential complex non-linear interactions amongst features and improves quality Features. Since MOOCs' data are log of time series, CNN as a prediction model cannot help to correctly predict students' dropout in MOOCs or evaluate their performance.

In some studies, some researchers merged the CNN approach as lower layers to extract features with recurrent neural networks (RNNs) while some others merged it with Long Short-Term Memory (LSTM) as upper layers that process input data sequentially to perform classification processes and achieve prediction [7,8]. However, there are still some shortcomings for models that are not implemented on datasets from different platforms and do not take class imbalance (classification error) into account. Class imbalance problem is caused by fewer occurrences of data for a particular class, which result in sub-optimal performance. Jordan [9] revealed that completion rates for MOOCs occur with a low percentage of 0.7% to 52.1% with a median value of 12.6%.

In spite of the need of a more accurate predictive model, literature reveals that previous predictive models are suitable for solving dropout problems but most of those models are based on feature engineering strategy. In addition, all those models perform equally in all dropout problems without considering class imbalance data. Since MOOC courses are available at any time, most students do not adhere to the actual-time of the course in their study. For instance, many samples of students spent less period of the course time but made good performance in the same way for the students who spent much more time of the course. Thus, the model ultimately considered them as dropouts because time steps were inconsistent. This problem falls under the name of class imbalance problem (classification error), which indicates that there are fewer occurrences of data for a particular class. This means that the false-negative rate is always high.

Accordingly, there are problems that have not been solved yet or which can be resolved better by employing new predictive models. Therefore, we proposed a novel predictive model, which can read directly logs data from any MOOC platform to automatically extract features. The significant features are selected as prediction features fed to a layer of prediction which deals with students' data as time series sequence. In addition, we proposed a custom loss function relying on a cost-sensitive technique by statistically calculating costs inferred according to students' activities and the actual days spent in the course with many samples to calculate varying misclassification costs for both false negatives and false positives and tune different weights for different categories and improve performance of dropout prediction.

Having been inspired by this approach, we merged CNN with LSTM, which automatically extract significant features and capture temporal-dependencies effectively in clickstream data as a predictive model to make the optimum prediction of dropout by dealing with class imbalance problem.

Our contributions addressed in this research are the following:

- The CONV-LSTM model deals directly with the raw clickstream data to automatically extract features by CNN Layers and select the significant features learned by CNN layers, which maximize classification impact.
- The LSTM model is employed to deal with the dataset as time series sequence, which mainly helps in the collection of students' interactive activities in sequence format after extracting important features of the raw data in CNN layers.
- To address class misbalanced problem by employing custom loss function to tune different weights for different categories.
- To assess efficiency of the proposed model, the experiment has been run on different datasets which have different structures from different platforms. Results of the experiments show that our model achieved results better than those acquired by feature engineering-based baseline methods.

The structure of the research is as follows. Section 2 offers background of what have been searched before about prediction in MOOCs. Section 3 presents the methodology utilized in the research including phases of model architecture. Study experiment and results of the analysis are provided in Section 4. Discussion and implications for research are given in Section 5. Finally, the main conclusions are highlighted in Section 6.

2. Related work

MOOCs have gained extensive attention as a new method of education since 2012 in social media and in the academic area. They have acquired a large number of enrollees and a vast amount of information stored about what happens during the course. This has sparked many researchers for further analysis. Breslow [10] investigated the first course MOOC, "Circuits and Electronics" (6.002x) on edX in terms of the time students' engagement in each course content, knowledge of students' background, factors influencing the final degree, etc., and suggested that students learning impact could be enhanced through log mining. He analyzed the relations between students' background, behavior learning patterns on the course 6.002x dataset on edX and their final performances. On the evidence of these studies, the researchers observed that most of the students did not complete their courses, and retention rate for MOOCs was very low. Retention of students was a significant measure for continuing the education process of MOOCs as those abided by the syllabus had the chance to achieve the anticipated educational gains from their learning expertise. In this respect, many researchers studied the likely reasons which may make students drop-out by analyzing several students' activity logs of the MOOCs. In Allione and Stein [11], the Cox proportion hazard model was used to analyze students' retention of the Coursera platform based on students' activities and their demographic data. According to the current literature, dropout in MOOCs has not a specific definition, and it was defined according to datasets and prediction purposes. Students' behaviors in an online learning environment trace the effects of researchers' interest in investigating the variables related to students' retention in such environment such as navigation behaviors and learning outcomes [12]. The relationship between modeling students' profiles in studies has been explained in the literature. Learning strategies have been reviewed by Kizilcec [12].

Results of the abovementioned studies revealed that learning tools with which students are in interaction are completely diverse, and this diversity may influence their learning outcomes. Besides, different prediction algorithms and models of extracting features were adopted to study students' behavior in online learning. Youssef et al. [13] indicated that the first traditional approach relies on generalized linear models such as logistic regression, linear SVMs, and survival analysis. Each model analyzes according to various types of behavior based on hand-crafted features extracted from raw data (e.g., clickstream, discussion forum, grades). In Amnueypornsakul et al. [14], features engineering was used to extract features from raw data, then an SVM with RBF kernel was employed to predict students' dropout. Fei & Yeung [15] noticed dropout prediction as a time series sequence prediction problem, and LSTM for prediction with considering various definitions of drop out was used. Many researchers employed Machine Learning models extensively studying early prediction to identify students who are at-risk based on their behavior [15–17]. For example, Logistic Regression, Decision Tree, K-NN, SVM, Multi-Layer Perceptron and Naïve Bayes Classifier were conducted. Chui et al. [1] studied risk and marginal university students then proposed a Reduced Training Vector-based support vector machine (RTV-SVM) algorithm to predict students' status. Results showed that accuracy of the predicted rate reached 91.2% for at-risk and marginal university students. Mubarak et al. [16] noticed that dropout prediction is a time series prediction problem. They used two models; the logistic regression by improving a regularization term, and the Input-Output Hidden Markov Model to analyze students' behavior and their interactive information in a virtual learning environment to early predict their dropping. Though these predictive models help in identifying early dropouts, another problem other than predicting students' dropping out emerges, which is predicting students' performance. Several studies are based upon considering data of MOOCs as time series on dropouts prediction. They take advantage of the power of deep learning methods in the field of learning analytics to predict students who are at-risk. Deep learning employs methods that incorporate developing a model containing multilayers to learn from raw data. Every layer transfers the representation to a more abstract pattern for the subsequent layer [18]. Although these methods accomplished desirable results for predicting students' dropouts, feature engineering is mostly used to extract features, and this takes a lot of effort on hand-crafting to make it acceptable to the model. It is a complicated and time-consuming process. Furthermore, feature engineering strategies may not be effective for a variety of MOOC datasets. These features inevitably lose influence on prediction models and also include unintentional biases due to researchers' subjectivity. Plus, many of these features are unfamiliar for the model, which requires us to transform feature extraction methods depending on the model. These features include video clickstream, assignment-related features, discussion forum, and activity-related features [19]. For the previous reason, many studies are required to solve these problems. Many researchers used techniques without manual feature engineering processes to predict dropout. They employed Deep neural networks (DNN) models since these were used to exceed feature engineering processes, especially CNN and LSTM models regardless of fields and raw data natural form as input [15,20]. Few previous works explored DNN model [21] and CNN followed by RNN [22]. Yet, all of these recent models have given sub-optimal performance. This is mainly due to class imbalance problem in datasets because there are occurrences of fewer data for a particular class. There are various approaches to manipulate class imbalance such as resampling data by oversampling or under-sampling [23]. Under-sampling may be applied to the majority class but may delete useful information. Oversampling can simply add unwanted noise with overfitting risks to the minority class. Another strategy is cost-sensitive technique that determines higher misclassification costs to the minority class[24].

Overall, other studies took advantage of the ability of deep learning models to better grasp students' behavior in online learning environments. Our study is one of the important studies that investigate to exhibit the strength of deep learning models to improve predicting students who might not complete the course or those who are not active in online learning environments.

3. Methodology

The present study was performed in multi stages as shown in Fig.1. (This will be explained in detail in a coming section.).

3.1. The proposed model

The architecture of our proposed CONV-LSTM model is exhibited in Fig. 1. It consists of the following components: data transformations, convolution and prediction phases.

3.1.1. Data transformations phase

In this phase, raw data is transferred into appropriate input data that can be learned by the next phase. Raw data are students' behavior in MOOCs courses, which are stored in a log file as text format which contains different information about students' activity in a chronological order. By analyzing activity records of students' learning in different MOOC platforms, there is no steady format for activity logs of students such as Coursera and edX platforms [10]. Logs' structure of each platform varies, but at least they generally contain (a student ID), the time, and event type. Therefore, it is possible to apply the second stage to these data directly in our model. We want to process them into a two-dimensional matrix which can be fed as input layers. Data transformation algorithm is proposed for this purpose according to the time and behavior dimensions of data records. Two-dimensional convolution is implemented on the clickstream data logs MOOCs based on the temporal and behavioral attributes. For each record, the event on the platform with a timestamp is recorded. Several adjacent timestamps form a time unit, which can be described as an hour (in our data) or a day, etc. Several time units form the time period (day), and each time unit has a behavior dimension that represents students' behavior logs. For each behavior dimension, the number of the different behaviors in the analogous time unit is counted as the value of this dimension that represents size of this dimension. Therefore, each time period (day) represents a two-dimensional matrix of time units and behavioral attributes. Algorithm 1 exhibits the transformation stage process. The outputs of a data transformation algorithm are concatenated matrices of times period, and every row forms a matrix of activities for time units that are equal in size. Thus, every student, who studies at least one course, has matrices of the size $T^0 \times V^0$, where T^0indicates the number of time units in every time period, V^0 represents matrix vectors length at every time unit. We employ it as input data for our model as indicated in Fig. 2.

3.1.2. The convolution phase

In order to make the model effective in solving the problem of prediction of dropout, we regard the properties of the inputs. For every time period, we set a matrix to represent events within it. To extract features from each of the matrices, we use CNN as a lower part. The input of CNN is spatial in nature i.e., it is a two-dimension input. It also uses the convolution operation, which is a feature extraction technique, which involves the employment of a special matrix known as kernel. Thus, the Convolution layer generates another output matrix which is proportional to the input shape. CNN also involves Max pooling which is similar to convolutions in the way they all extract information from the input's regional spatial regions to create a matrix. However, max pooling has no kernel linked with it. It simply produces the maximum number of gained windows across a 2D input as output matrix.

3.1.3. The prediction phase

Due to the fact that students' learning data are essentially time-series data, students' status will be affected at a determined time through their activities in the past. For this purpose, LSTM has the ability to model early learning behaviors with different time-series data. This means that LSTMs can manage long term dependencies much better than other traditional RNNs [25]. It includes a recursive loop that helps the model to include prior inputs with the current one. But the module of repetition has a varied structure, and instead of owning one neural layer, there are three gates called input, forget and output gate. Besides, another unit is the memory cell which dominates the data to be signed in the cell by running straight throughout the whole chain with some major linear interactions of the three gates. In addition, LSTMs can retain and connect previous information. The parameters of LSTM are trained by backpropagation through time processes (BPTT) [25]. Therefore, LSTM was selected as the upper part of our model. The LSTM outputs fid into the last

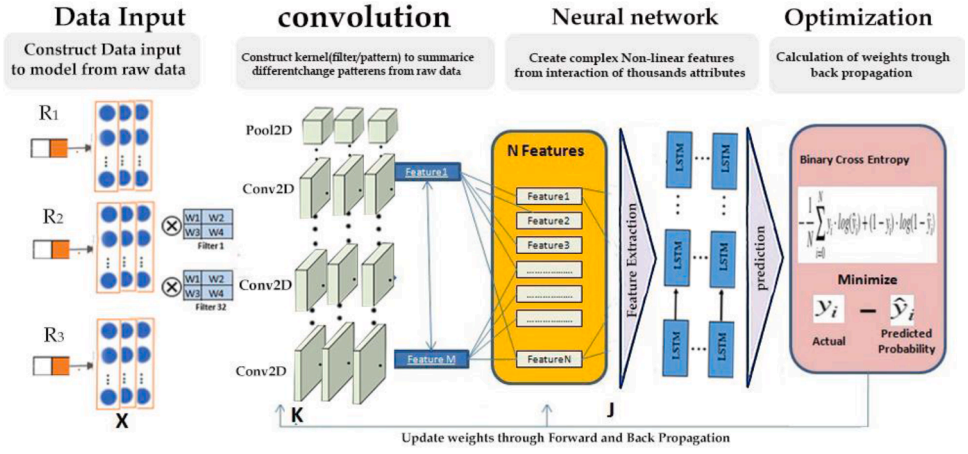

Fig. 1. Structure of CONV-LSTM.

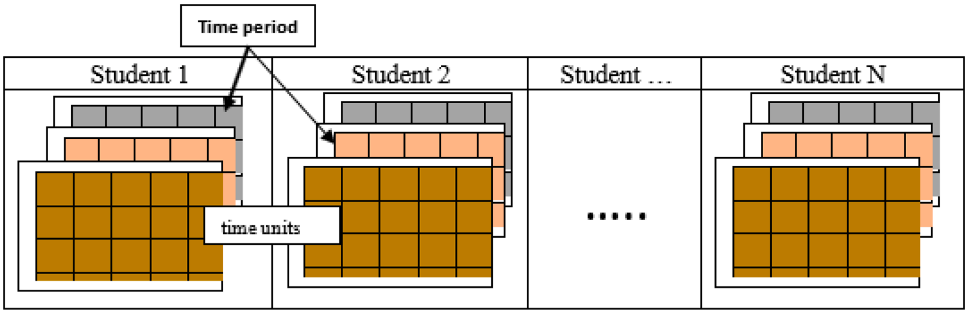

Fig. 2. Description of input data after the transformation process, where the period represents a matrix of time units.

layer (dens layer) of outputs, and it the output is set to a two-dimensional matrix. The "1″ is defined as "not dropout" and the "0″ is defined as "dropout".

3.1.4. Model architecture

In our model, CNN and LSTM are combined as shown in Fig. 1. The core of the model is three-layer two-dimensional convolution, one pooling layer and two-layer LSTM. A rectified linear unit (*ReLU*) function was employed in every hidden layer. Both the convolutional as well as the LSTM layers were trained together by using Backpropagation Through Time. The output of the LSTM last layer was passed into a layer dense with sigmoid function, which predicts the probability of future dropout relying on students' past behavioral clickstream data. One pooling layer was just used to maintain most of the feature information. As for this model, we represent a student as an instance, and every instance has many time period matrices as mentioned earlier. Therefore, model inputs are represented as N matrices $X_1,X_2,\ldots X_n.,X_N$, each of the size $T^0 \times V^0$.

The first three layers of the model are convolutional layers. To apply valid convolution, we set the value "valid" of the parameter "padding" and the stride is 1. For every instance, the inputs are as inputs of two-dimensional data$T \times F^{l-1}$, each of the size $N^{l-1} \times M^{l-1}$. So, a two-dimensional convolution kernel was used with kernel size $H^l \times D^l$ where $l$is the index of the layer for processing. Thus, the output from layer$l$ is $T \times F^l$ matrices, each of the size $N^l \times M^l$, and the output matrix size of a convolution layer was computed according to Eqs. (1,2):

$$W = \left(\frac{W - F + 2P}{S}\right) + 1 \tag{1}$$

where F is filter width, P is the padding, and S is the stride.

$$\begin{aligned} N^l &= \left(\frac{N^{l-1} - H^{l-1} + 2p}{s}\right) + 1 \\ M^l &= \left(\frac{M^{l-1} - D^{l-1} + 2p}{s}\right) + 1 \end{aligned} \tag{2}$$

The rectified linear unit (*ReLU*) activation function was employed in convolution layers to compute the output of every time unit according to Eq. (3):

$$X_j^l = Relu\left(\sum_{i \in F_j}\left(X_i^{l-1} * w_{ji}^l + b_j^l\right)\right) \tag{3}$$

where F_j represents matrix of input feature maps of time units of the *ith* layer, the X_i^{l-1} represents the feature map for output of the convolutional layer $l-1$ that is an input of the l layer. w_{ji}^l is a matrix used to generate the *ith* feature map in *lth* layer from *jth* feature map generated in the $l-1th$ layer. It is called convolution kernel's weight. b_j^l is the bias of the generated *jth* feature map in the *lth* layer. Operator "*" denotes convolution operation. X_j^l is feature map for each output of the convolutional layer l.

The fourth layer is the max-pooling layer whose function is to progressively reduce the computational cost by reducing the number of parameters and computation in the network. Here, the max-pooling operation is applied to every feature map. The output matrix is the most significant features (features with highest values) as Eq. (4).

$$X_j^l = Pooling_{max}\left(X_i^{l-1}\right) \tag{4}$$

where X_j^l are the feature maps generated by the max-pooling layer.

The first four layers have been employed to extracted features in adjacent time units. The features extracted are combined into a vector each time unit by a fully connected layer. All vectors form $V = (V_1,V, \ldots, V_n)$. Then, they are fed into the LSTM layer.

In the fifth and sixth LSTM layers, the LSTM layers deal with output matrix of CNN layers as time-series data. Eq. (5) below

describes the operation of two LSTM layers:

$$h_t^l = Relu\left(\sum_{i \in M_j} \left(V^t * w^l + h_{t-1}^{l-1} * U^l + b^l\right)\right) \tag{5}$$

where h_t^l represents the hidden state of layer t at time,w^l and U^l are weight matrices and b^l is the bias. The output h_t^l is fed into a fully connected layer. The final output is a problem of binary classification. Therefore, the sigmoid function was used in fully connected layer to present the final output, where we set "1″ as "not dropout" and "0″ as "dropout". Sigmoid function is defined by Eq. (6) as:

$$\sigma(z) = \frac{1}{1 + e^{-(Vh^t + b)}} \tag{6}$$

where h^t is the hidden state in time unit, it is a combination of all the information in the previous time units. T is matrix of the weight and b is the bias.

Adam optimizer and the custom loss function as a loss function were used in the model training process. The custom loss function was used depending on cost-sensitive learning during the training model to minimize the overall costs instead of minimizing misclassification. Further details are discussed in the next section.

3.1.5. Custom cost-sensitive loss function

A class imbalance problem means that the total of a certain class (positive) is far less than the total of the other class (negative) out of all the classes which one wants to predict. This problem is recognized as imbalance classification problems in machine learning field, which makes majority of the class dominate minority of the class. Thus, the model is mostly more biased to the majority class during the training phase, which results in un-optimal performance, which means that the generalization capability of the model relies on the training data.

Several strategies can handle imbalanced data problem such as resampling methods by oversampling the minority class or under-sampling the majority class [26]. Although the sampling-based method is one of the most effective approaches, over-sampling grows weight of the minority class by adding or composing new minority class samples. This causes noise with overfitting risks. Under-sampling performs decrease examples in the majority class to equal the minority examples. This can result in losing valuable information, which will be useless for the few amounts of data [23]. Another method is cost-sensitive learning, which designates various costs to the types of misclassification errors that can be made during training the model using specialized methods to take those costs into account, or a it is called a penalty correlated with the wrong prediction [24].

In the educational datasets, whereas some students took the entire course in a certain time period which was far less than the actual time, some others took the entire course in a certain time period which was longer than the actual course time, and they actually passed the course. This can frequently occur in MOOCs data, because they are available at any time. Therefore, the classifier is unable to accurately predict these instances and biases to the majority class because it deals with students' data as sequential data. The cost-sensitive learning category that takes the costs of prediction errors [26] was used to assign a higher cost during the training to impose the model to paying more consideration to these instances. The main aim lied in improving the standard loss function "binary cross-entropy" which can be represented by Eq. (7):

$$loss = y_{true} * \log\left(y_{pred}\right) + (1 - y_{true}) * \log\left(1 - y_{pred}\right) \tag{7}$$

Therefore, we need to handle the loss function with costs correlated with each of the four cases (False Positives (FP), False Negatives (FN), True Positives (TP) and True Negatives (TN) so that the model can learn to make optimum predictions accordingly. We first calculated cost value using statistics manner (students who have a quite small number of days and a quite large number of events and those who have a very large number of days and a very small number of events according to course days), according to Eq. (8):

$$cost = \begin{cases} \dfrac{\sum_{i=1}^{n} events}{(interaction\ days * cousre\ days)} & interaction\ days < 1/2 course\ days \\ \dfrac{\sum_{i=1}^{n} events}{(cousre\ days)} & interaction\ days > course\ days \end{cases} \tag{8}$$

We mostly focused on misclassification errors, the so-called False Positives and the False Negatives that probably gain the best attention. Thus, we assumed that each instance has some True Positive C_{TP}, True Negative C_{TN}, False Positive C_{FP}, and False Negative C_{FN} costs associated with it or a penalty which is correlated with an incorrect prediction. Therefore, we set the costs of True Positive and True Negatives which are equal and both costs are zero, False Positive has 0,0001 cost, and the cost of False Negative was computed by Eq. (8) which is previously mentioned. Thus, the custom loss function formula is as Eq. (9):

$$Custom\ loss = y_{true} * \left(C_{FN} * \log\left(y_{pred}\right) + C_{TP} * \log\left(1 - y_{pred}\right)\right) + (1 - y_{true}) * \left(C_{FP} * \log\left(1 - y_{pred}\right) + C_{TN} * \log\left(y_{pred}\right)\right) \tag{9}$$

C_{FP}, C_{FN}, C_{TP} and C_{TN} are the costs for False Positives, False Negatives, True Positives and True Negatives accordingly. Also, y_{true} is the likelihood of the truth value, and y_{Pred} is the likelihood predicted by the model.

The custom loss function leads to reducing the model cost on the training dataset, and operates with Adam optimizer throughout the training process. Algorithm 2 exhibits pseudocode for a custom loss function.

4. Experiments & evaluation of results

In this section we will presents the dataset used, the experiments executed on it and parameter tuning of model and interpretation of the results achieved by the employed model. Then evaluation of results of the model in the light of baseline methods follows to compare and discuss results of the model.

4.1. Dataset

Datasets of this research were collected from two different MOOC platforms in order to assess the effectiveness of the model on various platforms. The first dataset contains Five courses which were collected by the Center for Advanced Research Through Online Learning (CAROL)[1] at the University of Stanford. In these courses, 78,623 enrollment records and activities are registered as events in a log file up to 2038,690 event records. The dataset consists of activity records of the students about which course they are enrolled (e.g., events on a video, reading a text page, a quiz, or solving a problem, discussion) which include 22 events with timestamp of each event and other information about the courses. The second dataset comprises 39 courses provided by KDD Cup 2015[2] as a public dataset from the online course platform XuetangX. The dataset consists of course ID, student ID, students' activities which include seven different main types of activities ("problem", "video", "access", "wiki" "discussion", "navigate" and "view page") as events in a log file, the timestamp of each event, and other information about each course. This dataset includes 120,542 enrollment records in addition to 8157,277 records in the activity events file. The statistical data of both datasets is presented in Table 1.

For both datasets, the Record Period was 30 days for each course. Some irrelevant data are filtered out during preprocessing such as empty columns and events that happen before a student's formal registration (as a free-trial period). Data have been anonymized for the privacy protection of students' personal information, and students' identity is determined via unique ID and the course ID instead of the course name.

4.2. Experimental study

In order to assess the extent of effectiveness of the proposed model in predicting students' dropout and their performance, the problem was transformed into a binary classification task ("0″ dropout and "1″ non-dropout). The proposed model architecture produced optimal results by combining CNN and LSTM deep neural models and considered the behavior-temporal dependency features of students' data. Each layer in the model was fed from outputs of the previous layer as its input. During the training phase, the data were combined on the same day as a matrix. The input of every instance was 30 matrices. Every matrix was of size 30×34 as mentioned in the transformation stage. In three Convolutional hidden layers, 32, 32 and 24 filters and biases were used to produce feature maps respectively. All filters were of the size 5×5, and padding set "valid" and Stride was 1. The fourth layer was the Maxpooling with the size 2×2. The output was 30×12 matrices whereas the output layer was then flattened to be fed into LSTM layers. The proposed model has two hidden LSTM layers followed by a dense layer to produce the output between [0, 1], indicating whether a student is likely at risk of dropping out or his performance is good. During many epochs, the model was able to identify meaningful and certain low-level features in data and prediction using the sigmoid activation function in the last layer. Table 2 exhibits the details of the parameters in the proposed model.

We carried out the experiment many times with different parameter values in order to find the optimal parameters. Batch normalization between layers was applied to promote rapid learning during the in-depth learning process. The model was trained for 100 epochs, and early stopping was used to prevent overfitting. The optimization Hyperparameters of optimizers were tuned with the rate of learning as shown in table 3, and the rate of the dropout was between LSTM layers of 0.5.

The objective of training the model is to tune the correct weights for the network by multiple forward and backward repetitions, which ultimately tries to minimize binary cross-entropy (misclassification cost) by applying cost-sensitive to the loss function. The custom loss function in Eq. (9) used costs as a penalty for misclassification [24]. Where C_{FP}, C_{FN}, C_{TP} and C_{TN} are the costs for False Positives, False Negatives, True Positives and True Negatives accordingly. Also, y_{true} is the likelihood of the truth value, and y_{Pred} is the probability predicted by the model. The Keras framework with back-end as TensorFlow (*tf*) was employed. However, implementing custom loss function directly in Keras is complex because Keras does not allow arguments other than y_{true} and $y_{Pred}$to pass to the loss function. Thus, another function was used to wrap loss function to transfer the constant variables as mentioned in Algorithm 2.

Due to the character of the research problem, some instances are not-active but completed studying and passed the course (a classification error problem). Therefore, accuracy is not a suited evaluation metric. Thus, the AUC metric was used during training for the sake of evaluating the effectiveness of the model (When Keras doesn't have default AUC metric, we can leverage *tf* to build customized evaluation function for AUC). Algorithm 3 shows pseudocode for AUC function. Hence, binary cross-entropy presented proper results. For each dataset, 20% of the data was used for testing, and 15% of data was used for validation.

The results represented in Fig. 3 illustrate the efficiency of the proposed model by demonstrating the advancement in AUC results during the training process of both datasets by Amad optimizer, which achieved better performance than other optimizers. We implemented the model to both datasets to see how the performance of the model on datasets from different platforms was. Fig. 3 (a and b) shows that the AUC continuously increases from the initial training stages for both datasets, and it represents strength and the

[1] https://datastage.stanford.edu
[2] http://kddcup2015.com

Table 1
The statistical data of both datasets.

Item	Dataset11	Dataset2
Courses	5	39
enrollment records	78,623	120,542
Activity logs	2038,690	8157,277

Table 2
Hyperparameters used in the Proposed Model.

Propose Model Hyperparameters. Layer	Name Layer	Activation	Hyperparameters
1	Input	–	
2	CNN2D_1	*ReLU*	Filters:32, Filter size,5 × 5
3	CNN2D_2	*ReLU*	Filters:32, Filter size,5 × 5
4	CNN2D_3	*ReLU*	Filters:24, Filter size,5 × 5
5	Maxpooling2D	–	Pool size,2 × 2
6	LSMT_1	Tanh	20 units
7	LSTM_2	Tanh	20 units
8	Dens	Sigmoid	2 classes

Table 3
Hyperparameters of Optimizers.

Optimization Hyperparameters. Num	Name Optimizer	Learning Rate
1	Adam	0.005
2	SGD	0.005
3	RMSprop	0.0005
4	Adagrad	0.0001
5	Adadelta	0.05
6	Adamax	0.0001
7	Nadam	0.0001

extent of the model to extract significant features and accurate prediction of input instances. It also learns the classification pattern of both classes. With the collected events logs of every student, the model predicts whether a student is anticipated to drop out of a course with the AUC scale up to 85% and 86% in both datasets respectively. The decreases in the loss worth of the model can be observed in Fig. 3 (a, b), which should be naturally reduced. This can refer to the variation between the true value and the class label outcome from the proposed model, that showed extent of the effectiveness of our custom loss function.

Due to our focus on solving class imbalance problem, the precision-recall curve employed more evaluation. Fig. 4 demonstrates the precision-recall curve of the model during the prediction phase for both testing datasets. A precision-recall curve is a plot of the precision (y-axis) that points out the ratio of the accurately classified instances i.e., predicting dropout from all the instances predicted and the recall (x-axis) that illustrates the effectiveness of the model precisely identifying the dropout from all dropout instances in the data. Moreover, the Area under the Precision-Recall Curve (AUPRC) denotes a different tradeoff between the true positive rate and the positive predictive value for a model. We also note in Fig. 4 that the AUPRC is 78% and 80% at both datasets respectively, insinuating the robust performance of the model.

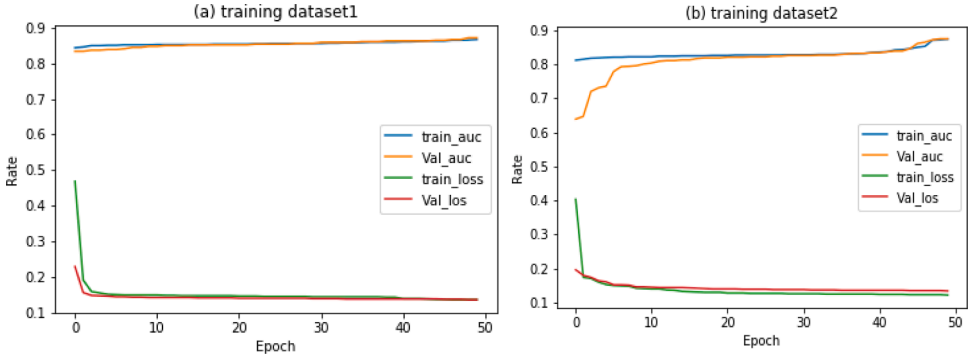

Fig. 3. The proposed model training for both datasets.

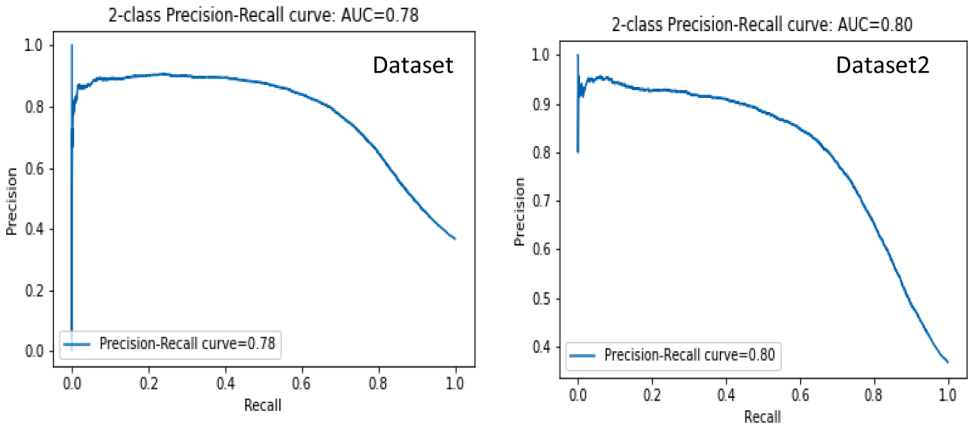

Fig. 4. Precision-Recall cure for the Proposed model.

To further check the performance of the model, table 4 manifests the optimizer-wise results of the proposed deep model trained with or without custom loss function. The proposed model has the same setting with initial random weights set to the default values of each layer on both states. They were trained on different datasets, and thus predictions were made. Results showed that the proposed model with a custom loss function achieved high performance better than as if it were without this function for all optimizers with the highest F1-score. The Precision of Adam optimizer on the dataset 1 achieved 96%. Compared to Adadelta and Adamax, SGD achieved 92% and 85% Precision and Recall improvement in dataset 2 respectively. The model trained by using RMSprop optimizer achieved F1- score 86%, Precision 95%, and Recall 80%. Adam reports 91% F1- score, 96% Precision and 87% Recall proved as top optimizer on proposed model with custom loss function compared to other optimizers. On the other hand, results showed that the proposed model without a custom loss function achieved desirable performance, as we noted within the right side in table 4. The Precision of Adam optimizer on dataset 1 reaches 94% followed by RMSprop, Adagrad and Adamax achieved 93%. The model trained using SGD optimizer achieves F1-score scoring 87%, Precision 90%, and Recall 84% in both datasets. Adam reports again proved as top optimizer on the proposed model compared to other optimizers.

Besides, the ROC curve was used to evaluate performance of optimizer-wise of the proposed deep model trained with custom loss function as illustrated in Fig. 5. Results showed that the proposed model with a custom loss function achieved high performance better with Adam optimizer. The Roc of Adam optimizer on the both datasets achieved 86% followed by SGD (84%). Compared to RMSprop, Adadelta and Adamax achieved 83%, 80% and 83% respectively. Moreover, results proved that the proposed model achieve high performance in predicting students' MOOC dropouts with custom loss function and Adam optimizer.

4.3. Evaluation model with the baseline

In order to assess the extent of strength of our model regarding identifying students who are at risk of dropping, our model was compared to deep neural networks (DNNs), Support Vector Machines (SVM) and Logistic Regression (LR). These methods were chosen as a result of their repeated utilization in research society as baseline models [2]. These methods apply feature engineering to extract features. Therefore, we extracted 40 features from the raw records using feature engineering technique. For every student, a one flat feature vector forming of the collate values of features was fed to baseline models. Also, our model without custom loss function was

Table 4
Results of Proposed model with and without Custom loss function by multi optimizers on both datasets.

Dataset	Proposed model with custom loss F.				Proposed model with custom loss F.		
	Optimizer	Precision	Recall	F1 score	Precision	Recall	F1 score
Dataset 1	Adam	0.96	0.87	0.91	0.94	0.83	0.88
	SGD	0.91	0.85	0.88	0.90	0.84	0.87
	RMSprop	0.94	0.79	0.86	0.93	0.78	0.85
	Adagrad	0.95	0.79	0.86	0.93	0.78	0.85
	Adadelta	0.94	0.79	0.86	0.92	0.76	0.83
	Adamax	0.95	0.78	0.86	0.93	0.76	0.84
	Nadam	0.93	0.81	0.87	0.92	0.75	0.83
Dataset 2	Adam	0.96	0.85	0.90	0.94	0.85	0.89
	SGD	0.92	0.86	0.89	0.89	0.84	0.87
	RMSprop	0.95	0.80	0.87	0.92	0.77	0.84
	Adagrad	0.95	0.80	0.87	0.93	0.77	0.84
	Adadelta	0.94	0.78	0.85	0.92	0.77	0.84
	Adamax	0.95	0.78	0.86	0.93	0.76	0.84
	Nadam	0.94	0.77	0.85	0.91	0.79	0.85

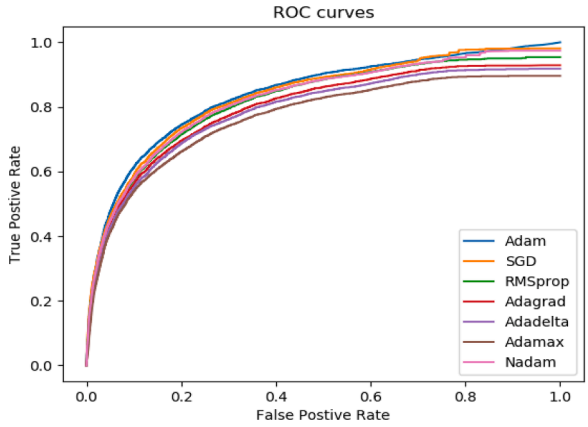

Fig. 5. AUC Scores of the Proposed Model through optimizers for two datasets.

applied for the purpose of dropout prediction, which we endeavor to show up the contribution of the custom loss function in our model. Due to our interest in addressing the classification error problem, the Precision, Recall, F1-score and the Area under the ROC Curve (AUC) score were used as evaluation metrics. The AUC is the value under the ROC curve over the interval [0; 1]. It answers the question "What is the maximum probability which the model can correctly distinguish whether the student falls as dropout or non-dropout?". Due the nature of the problem in this research, the accuracy metric was ignored. Table 5 shows a summary of the statistics collected from baseline models with our model representing "Dataset1" (top) and "Dataset2" (down as noted in the table for both datasets. Our model with custom loss function achieves a fairly comparable performance to task-specific baseline models in terms of F1 score and AUC. Moreover, results indicates that the custom loss function played an important role in minimizing the total cost (This means that it treated classification error problem).

As exhibited in table 5, the AUC of our model is relatively more than the baseline models, and it has a maximum value of 0.86 in both datasets. On the contrary, the SVM model gains a minimum AUC of 0.77 in "dataset1" and 0.76 at "Dataset2". One potential explanation of this is that the proposed model achieves a better class balance in "Dataset1".

Overall, the proposed model automatically extracts significant features and monitors the sequential behavior of each instance, because LSTM has a memory cell and also limits classification error by custom loss function whereas baseline models are incapable of achieving that. As mentioned earlier in this research study, the inputs of baseline models form a flat vector holding all the aggregated feature worth for every student all in one time. Accordingly, this handicaps the model to learn the sequential behavior of a student. Therefore, baseline models may incorrectly predict a student who is not active throughout a course, or they may incorrectly classify students. Moreover, the proposed model outperformed in terms of prediction accuracy when it is compared to baseline models.

5. Discussion of results and implications for research

MOOCs have recently become more popular and attractive for students due to their various qualities of being fully open and online, not like traditional education systems. However, dropout rate in MOOCs is very high and completion rates rarely ever exceed 25% for highly committed students [9]. In this respect, attrition has become one of the key research challenges for the educational community, which means that most students, who have enrolled, wish to know the course content without any motivation of completing the course [15,17]. This problem has led many researchers to optimize predictive models that can be used to achieve a better understanding of the learning abilities, not only for the case of learning results but also to detect students who are at-risk of dropout and make interventions that could drive to better completion rates.

Some predictive models are based on features engineering to extract features from students' records. Each model considers several

Table 5
Comparison of the proposed model with baseline models.

	Model	Precision	Recall	F1-Score	AUC
Dataset1	CONV-LSTM with custom loss Function.	0.91	0.88	0.89	0.85
	CONV-LSTM without custom loss Function.	0.90	0.87	0.88	0.84
	DNN	0.95	0.82	0.88	0.78
	SVM	0.94	0.82	0.88	0.77
	LR	0.97	0.77	0.86	0.79
Dataset2	CONV-LSTM with custom loss Function.	0.93	0.87	0.90	0.86
	CONV-LSTM without custom loss Function.	0.91	0.82	0.86	0.84
	DNN	0.96	0.84	0.89	0.80
	SVM	0.95	0.85	0.90	0.76
	LR	0.97	0.79	0.87	0.77

Algorithm 1
Data transformation phase Algorithm.

ALGORITHM 1: Data transformation phase Algorithm.

Input: let student set S, Course set C, and event set E.
Output: the dataset D with two-dimensional and each element as matrix form *input data for model.*

```
foreach course C_i ∈ C do
    Get events set E_i ∈ E about c_i
    Find course c_i 's time units T_ci
    Get students set S_i ∈ S of c_i
    foreach student s_il ∈ Si do
        foreach course time T_ci do
            Get events set E_il ∈ E_i of s_il
            Get the start unit time T_s and the end unit time T_e of E_il
            For (j=T_s; j<=T_e; j++)
                Calculate each event E_il with same timestamp j
                Set event count as vector v
            End
            Set M_Si = M_Si ∪ v
            Set events s_il as matrix M_Si
        End
    End
    Set D = D ∪ {(ci , si , M_Si )}
End
```

Algorithm 2
The custom loss function Algorithm.

Input: custom loss (C_{FP}, C_{TP},C_{TN},C_{FN})
def loss_function(y_true, y_pred):
$Customloss = y_{true}*(C_{FN}*\log(y_{pred}) + C_{TP}*\log(1 - y_{pred})) + (1 - y_{true})*(C_{FP}*\log(1 - y_{pred}) + C_{TN}*\log(y_{pred}))$
return loss_function

Algorithm 3
The AUC function Algorithm.

Inputs: y_{true}, y_{Pred}
AUC = *tf.metrics.auc*(inputs)
K.get_session().run(tf.local_variables_initializer())
return auc

types of behavioral and predictive features extracted from different raw activity records. The features engineering process construct new features from raw data in a heuristic method, therefore, the features are extracted manually. This means that researchers determine the nature of the features extracted (bias by researchers), based on domain knowledge, inspiration, and data manipulation. Some researchers consider the features related to demographic features of the student (e.g., age, country of origin, level of schooling, original language, profession state, etc.) as prediction features to build predictive models, which can be extracted from the platform log or can be collected by surveying students' perspectives [17]. On the other hand, other researchers consider video-related features which occur when students interact with videos on the platform as features, while others believe that Forum-related features and variables related to Platform can be used as features[19]. However, the constructed prediction models have significant limitations in dealing with the massive nature of MOOCs data for the variety of characteristics they have when students have various environments (in terms of education, culture, personality, and so on). Also, there can be several various behaviors which are considered as a challenge to the extraction of features that can be very relevant to the efficacy of prediction models. As suggested by Moreno-Marcos [5], extraction of features from data can sometimes be even more important than algorithms because features require to capture suitable information in relation to the feature to be predicted. However, this suggests the question about which features are efficient to predict. When some features are lost, then what is the extent of the strength of the predictive model with those present features? Based on the limitations of the construction of the available predictive models construction in MOOCs, our model benefited from the strength of CNN to automatically extract features from various raw activity records, which extract many relevant features out of a dataset from which the best can be selected and used for modeling. These features can be interpretable and can limit data leakage by employing dimension reduction/feature selection method at any time to get rid of redundant/zero important features. Besides, features extraction process is not affected by the researcher's bias, plus it does not need numerous iterations of features engineering, and it does not also

consume the time spent for features engineering. The prevailing general aim of this research is to utilize the aggregation of various types of features in order to obtain prediction strength and take features from different sources whenever possible. Therefore, features employed in constructing the model directly influence the results obtained. Thus, prediction does not include interactions with only specific features but also with various features which influence each other and are related to the obtained outcomes. In this research, predicting students who are at risk of dropout MOOCs is anchored to a set of diverse features extracted by the CNN model. Since students' data in MOOCs are considered as sequential classification, the LSTM model is merged with CNN as a high layer to model students' behavior and obtain high predictability of students' performance. It takes into consideration the influence of the sequential relationship of students' behavior of dropout. Thus, the early prediction and the classification of students who are at risk of dropping out of their courses provide an opportunity to facilitators of the MOOC to give support resources which will enhance students' self-confidence and increase the instructor/student communication. In other words, predicting students' classification in the course will enable instructors to determine their interventions according to each student's profile. Despite the importance of identifying early dropouts, some students who do not drop out of the course are non-active (passive-active) and eventually pass. Most predictive models are unable to track them but can classify them as dropouts. In order to also detect this status in order to make interventions, our model takes into consideration the influence of class imbalance on prediction by setting cost-sensitive learning categories for different classifications based on students' activities and the actual days spent in the course. While carrying out the experiments, we noted that the predictions are not sometimes consonant with truth values even when a student (who is not active) performs well, and the prediction of that student's performance across course days decreases. Besides, the predicted performance through time-steps is not constant. This issue was addressed by adding cost-sensitive to the loss function addressing the penalty correlated with the incorrect prediction. This ensures that our model is accurate for all prediction classes, and it increases the performance of dropout prediction. Referring to the results mentioned in table 5, the CONV-LSTM with the custom loss function perform better than CONV-LSTM without custom loss function and other baseline models, which indicates that the cost added to the loss function has improved the predictive ability of the model.

Furthermore, this research study addresses the crucial issue of dropout in MOOCs. This research contributes to presenting the efficacy of DNN methods for devising data-driven decision-making policies, addressing concerns encountered by students and thus assisting educational institutions to keep their learning process alive. Besides, the results gained can be employed to develop pedagogical guidelines for important decisions. Prediction of students' performance will allow instructors and decision makers to adopt a realistic approach for timely interventions in order to positively guide students by providing suitable suggestions and counseling. In order to generalize the results of this study, a more experimental implementation is required on datasets from different platforms paying special attention to the qualitative features of prediction. This needs to update the basic assumptions of the research to enrich the proposed model with factors which interpret students' behavior.

6. Conclusion

The present study contributes to optimizing the identification of students who are at risk of weak performance. The proposed model deals directly with the raw clickstream data and automatic extraction of important feature, which saves much time and efforts, unlike machine learning-based dropout prediction methods which need manual feature engineering. Our model takes into consideration class imbalance problem and employs the custom loss function which introduces different weights for different classes and reduces total misclassification costs of the model and improves the performance of prediction of dropout. The experiment was conducted on different datasets from different platforms. Results show that the proposed model outperformed baseline methods relying on feature engineering performed by specialists and CONV-LSTM model without custom loss function.

Overall, results illustrate the effectiveness of the proposed model in evaluating the prediction of students' performance, which can support instructors and stakeholders to formulate a learning analytics framework and can also participate in decision-making to achieve corrective strategies to support and guide students in order to help them continue course and be productive. For future research directions in this field, we may determine interaction patterns of students by deeply studying students' data. We may also determine which activities may have an important impact on the performance of students by mining textual data, and may also consider class imbalance problems. All this will enable us to formulate different patterns for students to determine which certain performance category they should belong to, and this can also help educational stakeholders to promote necessary pedagogical guidelines for students' retention.

Availability of data and materials

The first dataset that supports the findings of this study is available in the Center for Advanced Research Through Online Learning (CAROL) at the University of Stanford Https://Iriss.Stanford.Edu/Carol, but there are some restrictions applied to the availability of these data, which were used under license for the current study, therefore, they are not publicly available. Data are, however, available with the authors upon reasonable request and with permission of CAROL at the University of Stanford. The second dataset is available to the public at http://kddcup2015.com.

Funding

This work was supported by Shaanxi Normal University Graduate Education and Teaching Reform Research Project "Research on Cultivation of Graduate Innovative Practice" [grant number GEPR-20–43], the National Natural Science Foundation of China [grant

number 61,907,029] and Natural Science Foundation of Shaanxi, China [grant number 2020JM-307]. This work was supported by the Research Supporting Project Number (RSP-2021/389), King Saud University, Riyadh, Saudi Arabia.

Authors statement

Ahmed A. Mubarak School of Computer and Science- Shaanxi Normal University-Xian- China, 710,119. Department of Computer, Ibb University, Ibb, Yemen. Han Cao School of Computer and Science- Shaanxi Normal University-Xian- China, 710,119. Ibrahim M. Hezam Statistics and Operations Research Department, College of Sciences, King Saud University, Riyadh, Saudi Arabia Department of Computer, Ibb University, Ibb, Yemen.

Authors contributions

A. M., Conceptualization of this study, Data curation, Writing - Original draft preparation, Methodology and Software. C.H., Conceptualization of this study, Revision of draft preparation. I. H., Validation of methodology and results, Revision of draft preparation. All authors read and approved the final manuscript

Declaration of Competing Interest

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

Acknowledgments

We would like to thank the editors of the journal as well as the anonymous reviewers for their valuable suggestions that make the paper stronger and more consistent.

References

[1] Chui KT, Fung DCL, Lytras MD, Lam TM. Predicting at-risk university students in a virtual learning environment via a machine learning algorithm. Comput Human Behav 2020;107:105584. https://doi.org/10.1016/j.chb.2018.06.032.
[2] Waheed H, Hassan SU, Aljohani NR, Hardman J, Alelyani S, Nawaz R. Predicting academic performance of students from VLE big data using deep learning models. Comput Human Behav 2020;104:106189. https://doi.org/10.1016/j.chb.2019.106189.
[3] Xing W, Du D. Dropout Prediction in MOOCs: using Deep Learning for Personalized Intervention. Journal of Educational Computing Research 2019;57:547–70. https://doi.org/10.1177/0735633118757015.
[4] Wise AF, Cui Y, Jin WQ, Vytasek J. Mining for gold: identifying content-related MOOC discussion threads across domains through linguistic modeling. Internet and Higher Education 2017;32:11–28. https://doi.org/10.1016/j.iheduc.2016.08.001.
[5] Moreno-Marcos PM, Alario-Hoyos C, Munoz-Merino PJ, Kloos CD. Prediction in MOOCs: a Review and Future Research Directions. IEEE Trans Learn Technol 2019;12:384–401. https://doi.org/10.1109/TLT.2018.2856808.
[6] Qiu L, Liu Y, Hu Q, Liu Y. Student dropout prediction in massive open online courses by convolutional neural networks. Soft comput 2019;23:10287–301. https://doi.org/10.1007/s00500-018-3581-3.
[7] Wu N., Zhang M., Zhang L., Sun X., Gao Y., F.e.n.g.J. CLMS-Net: Dropout prediction in MOOCs with deep learning. ACM International Conference Proceeding Series, 2019, p. 1–6. 10.1145/3321408.3322848.
[8] Wang W, Yu H, Miao C. Deep model for dropout prediction in MOOCs. In: ACM International Conference Proceeding Series. ACM Press; 2017. p. 26–32. https://doi.org/10.1145/3126973.3126990. Part F1306.
[9] Jordan K. Massive open online course completion rates revisited: assessment, length and attrition. International Review of Research in Open and Distance Learning 2015;16:341–58. https://doi.org/10.19173/irrodl.v16i3.2112.
[10] Breslow L, Pritchard DE, DeBoer J, Stump GS, Ho AD, Seaton DT. Studying Learning in the Worldwide Classroom: research into edX's First MOOC. Journal of Research & Practice in Assessment 2013;8.
[11] Allione G, Stein RM. Mass attrition: an analysis of drop out from principles of microeconomics MOOC. Journal of Economic Education 2016;47:174–86. https://doi.org/10.1080/00220485.2016.1146096.
[12] Kizilcec RF, Pérez-Sanagustín M, Maldonado JJ. Self-regulated learning strategies predict learner behavior and goal attainment in Massive Open Online Courses. Computers and Education 2017;104:18–33. https://doi.org/10.1016/j.compedu.2016.10.001.
[13] Youssef M, Mohammed S, Hamada EK, Wafaa BF. A predictive approach based on efficient feature selection and learning algorithms' competition: case of learners' dropout in MOOCs. Education and Information Technologies 2019;24:3591–618. https://doi.org/10.1007/s10639-019-09934-y.
[14] Amnueypornsakul B, Bhat S, Chinprutthiwong P. Predicting Attrition Along the Way: the UIUC Model. In: Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs; 2015. p. 55–9. https://doi.org/10.3115/v1/w14-4110.
[15] Fei M, Yeung DY. Temporal Models for Predicting Student Dropout in Massive Open Online Courses. In: Proceedings - 15th IEEE International Conference on Data Mining Workshop. 2016. ICDMW; 2015. p. 256–63. 10.1109/ICDMW.2015.174.
[16] Mubarak A.A., Cao H., Zhang W. Prediction of students' early dropout based on their interaction logs in online learning environment. Interactive Learning Environments 2020. 10.1080/10494820.2020.1727529.
[17] Al-Shabandar R, Hussain A, Laws A, Keight R, Lunn J, Radi N. Machine learning approaches to predict learning outcomes in Massive open online courses. In: Proceedings of the International Joint Conference on Neural Networks; 2017. p. 713–20. https://doi.org/10.1109/IJCNN.2017.7965922. 2017- May.
[18] Lecun Y, Bengio Y, Hinton G. Deep learning. Nature 2015;521:436–44. https://doi.org/10.1038/nature14539.
[19] Nagrecha S, Dillon JZ, Chawla NV. MOOC dropout prediction: lessons learned from making pipelines interpretable. In: 26th International World Wide Web Conference; 2017. p. 351–9. https://doi.org/10.1145/3041021.3054162. WWW 2017 Companion, 2019.
[20] Whitehill J., Mohan K., Seaton D., Rosen Y., Tingley D. Delving deeper into MOOC student dropout prediction. ArXiv 2017.
[21] Kim BH, Vizitei E, Ganapathi V. GritNet: student performance prediction with deep learning. arXiv. International Educational Data Mining Society 2018.
[22] Sun D, Mao Y, Du J, Xu P, Zheng Q, Sun H. Deep learning for dropout prediction in MOOCs. In: Proceedings - 2019 8th International Conference of Educational Innovation through Technology, EITT 2019. IEEE; 2019. p. 87–90. https://doi.org/10.1109/EITT.2019.00025.
[23] Oquab M, Bottou L, Laptev I, Sivic J. Learning and transferring mid-level image representations using convolutional neural networks. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 2014. p. 1717–24. https://doi.org/10.1109/CVPR.2014.222.

[24] Shen W, Wang X, Wang Y, Bai X, Zhang Z. DeepContour: a deep convolutional feature learned by positive-sharing loss for contour detection. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 2015. p. 3982–91. https://doi.org/10.1109/CVPR.2015.7299024. 07-12-June.
[25] Zia T., Zahid U. Long short-term memory recurrent neural network architectures for Urdu acoustic modeling. vol. 22. 2019. 10.1007/s10772-018-09573-7.
[26] He H, Ma Y. Imbalanced learning: foundations, algorithms, and applications. wiley; 2013. https://doi.org/10.1002/9781118646106.

Ahmed A. Mubarak received Ms. degree in Computer Science and information from Menoufia University, Menoufia, Egypt. Currently, he is a Ph.D. candidate in Department of Computer Science in Shaanxi Normal University, Xi'an, China. He works as a lecturer in Education and Computer Sciences faculty, Ibb University, Yemen. His-main research interest is Machine learning, predictive modeling and Learning Analytics.

Han Cao is a professor at Shaanxi Normal University, and has taken charge of several Natural Science Foundation (NSF) projects of China. She is responsible for emerging engineering education research and practice research projects of Education Ministry of China, Science and Technology Plan of Shaanxi Province, China. Her research fields are knowledge graph, Big data analysis and tourism intelligence.

Ibrahim M. Hezam received a Ph.D. degree in operations research and decision support from Menoufia University, Egypt. He was a postdoctoral fellow in Industrial Engineering department at Pusan National University, Busan, South Korea 2018/2019. He is currently an assistant professor of operations research with king Saud University, KSA. His-research fields are artificial intelligence, operations research and decision support.