

MOOC performance prediction by Deep Learning from raw clickstream data

Gábor Kőrösi ¹ and Richard Farkas ²

¹² Institute of Informatics University of Szeged, Szeged, Hungary
korosig@inf.u-szeged.hu , rfarkas@inf.u-szeged.hu

Abstract. Student performance prediction is a challenging problem in online education. One of the key issues relating to the quality Massive Open Online Courses (MOOC) teaching is the issue of how to foretell student performance in the future during the initial phases of education. While the fame of MOOCs has been rapidly increasing, there is a growing interest in scalable automated support technologies for student learning. Researchers have implemented numerous different Machine Learning algorithms in order to find suitable solutions to this problem. The main concept was to manually design features through cumulating daily, weekly or monthly user log data and use standard Machine Learners, like SVM, LOGREG or MLP. Deep learning algorithms could give us new opportunities, as we can apply them directly on raw input data, and we could spare the most time-consuming process of feature engineering. Based on our extensive literature survey, recent deep learning publications on MOOC sequences are based on cumulated data, i.e. on fine-engineered features. The main contribution of this paper is using raw log-line-level data as our input without any feature engineering and Recurrent Neural Networks (RNN) to predict student performance at the end of the MOOC course. We used the Stanford Lagunita's dataset, consisting of log-level data of 130000 students and compared the RNN model based on raw data to standard classifiers using hand-crafted cumulated features. The experimental results presented in this paper indicate the RNN's dominance given its dependably superior performance as compared with the standard method. As far as we know, this will be the first work to use deep learning to predict student performance from raw log-line level students' clickstream sequences in an online course.

Keywords: RNN, MOOC, clickstream sequence analysis.

1 Introduction

The approximate number of Massive Open Online Courses (MOOC) students who enrolled or took part in a single or more courses is around one hundred million students [34]. This means that MOOCs can be considered as the prime method of knowledge acquisition online, their main advantage being the fact that neither geographic location, not monetary issues play a role in opting for this type of education. [7] As the MOOCs became quite popular among students, they sparked a great deal of research interest in MOOC data analytics. [8]

For all their benefits, the quality of MOOCs has been the target of criticism [9,10,12,16]. Almost all research has pointed to their low completion rates (below 7-10 percentage on average) as a property preventing more widespread adoption of these courses [13,10]. Stakeholders would benefit from knowing whether or not a given student was expected to complete the course, especially in view of the low completion rates. To solve these problems there has been two new research fields established: 1. Educational data mining [18], 2. (online) learning analytics [17]. Within the fields of learning analytics and educational data mining, we are able to create automated MOOC „dropout detectors" or „forecast performance" [5,16].

In terms of log data collection in the form of clickstream or social network measures, the MOOC systems offer a treasure-trove of data. The system gathers this data when students watch the various video lectures, try their hands at quizzes and interact with their peers in the available forums, discussing the learning materials during the course. [12]. The MOOC log data can be leveraged for prediction tasks through Machine Learning approaches [21]. Eventually efficient student models were created which would serve as a forecasting tool for estimating how many students were likely to drop out, or preferably complete the course. This was made possible by extensive research into comprehending and hopefully increasing the registration and completion rate, ultimately contributing to a better all-round learning experience in MOOCs.[20]. These issues involved the application of different supervised machine learning approaches so as to obtain an estimation for future learning results in MOOCs [7,31]. The majority of conventional methods is primarily based on generalized linear models, incorporating logistic regression, linear SVMs and survival analysis. Every model takes into account various kinds of behavioral and predictive characteristics gleaned from a number of raw activity records, such as clickstream, grades, forum, and final grades.[15].

Another solution is the Deep Learning (Deep Neural Networks) which could algorithmically find structure in log data and carry out prediction on various tasks in MOOCs [17,19].

However, the promise of Deep Neural Networks is to learn the temporal context of input sequences in order to make better predictions, only a few prior works explore this opportunity. Our extensive literature survey indicates that all studies in Educational Data Mining following the Deep Learning approach used cumulated data (daily, weekly, etc.) of feature engineering [8,10,14,13,16,21,32]. We didn't find any research where the input of the Deep Learning model was the raw log-line-level activity data.

The main contribution of our study is to investigate this challenge, i.e. whether Deep Neural Networks can benefit by using directly raw log data, rather than hand-designed statistics as commulated features. In this paper, we use log-line-level data and Recurrent Neural Networks (RNNs) to predict student performance at the end of the MOOC course as both a multiclassification and a regression task. The Recurrent Neural Network models each element of an activity log sequence (line by line), from the beginning of student's activity until a certain point in time, and predicts the student's final performance at the very end of the MOOC.

To the best of our knowledge, this will be the first work to use RNNs to predict user performance from log-line level students' clickstream sequences in an online course.

The rest of the paper is organized as follows: In Section I and II, we begin with the design of the problem, which involves the different hypothesis we make in the problem definition. Section III describes the main workflow, feature extraction, RNN-GRUs, and the prediction model especially. In addition, Section III. details the key technical points of the research and steps to implement it. Section IV details the key technical points of the research and steps to implement it. Afterward, we specify the details of the datasets and experimental setup we used in Section IV. Section V presents the experimental results of the prediction models on datasets provided by Stanford Lagunita's MOOCs. Finally, Section VI presents our key findings and conclusions.

2 Background

Specific fine-tunings were made by stakeholders during the course of education so as to be able to deal with both low MOOC outcome results considerable dropout rates. These modifications included email reminders to students or offering constructive feedback to specific learners at risk of leaving the course [14].

While this may sound like a workable solution, indeed, beneficial to the students themselves, given the reality of learner numbers of over ten thousand at a time, it could hardly be implemented in practice. Out of those 10,000 students there are probably an estimated 9,000 learners who would need such online support lest they drop out [10]. To address this issue, researchers recommend an automated system capable of reliably predicting the students' future performance in real-time [9,15].

The fact that the system is automatic, basing its work on learning analytics, makes it possible to both monitor and recognize those students who are in danger of leaving the course. At the same time, it will also be able to support early intervention design [10]. Hence, there is a considerable and ever-increasing amount research available about predictive modeling in MOOCs, especially focusing on models on the likeliness of a given student's dropout, stop out, or overall failure of completing a MOOC.

There have been earlier studies on student outcome forecast which were based on a wide range of characteristics obtained from clickstream data and the natural language used during postings in discussion forums, social networks, and assignment grades and activity [20]. These works relied on trace data from the introductory week or other specific times which would then serve as the basis for predicting students' outcome by way of the created prediction models [10]. Such prediction method enables the efficient detection of whether or not specific students are likely to drop out in the initial education phase, which, however, requires considerable time for feature extraction. The most popular feature representations are the measures of the distances among log events (time, points, etc.), aggregating the clickstream logs on a weekly basis and/or applying Natural Language Processing (NLP) on discussion forum content.

In terms of Machine Learning architecture, most prior approaches have used generalized linear models (including linear SVMs), survival analysis (e.g., Cox proportional hazard model), and Logistic Regression [21,14]. From another perspective, scientists compare classification against regression approaches [22]. For instance, J. He et al. (2015) [23] dealt with the Support Vector Machine (SVM) and Least Mean Square

(LMS) algorithms in order to identify what the learners' dropout rates were or how well they were performing in the MOOCs during the course period. Work was also conducted on clustering techniques, in which students were put into groups, clustering them on the basis of their student behavioral patterns. [24].

The paper of Jo et al. (2018) [25] reviews the variety of solutions in existing studies on modeling student behaviors via clickstream logs. It categorizes the solutions into top-down and bottom-up. Top-down approaches predefine a set of student behaviors of interest, such as disengagement and sequential navigation, and corresponding click patterns. These approaches provide interpretability, but analyses are focused only on pre-defined behaviors and patterns. In contrast, bottom-up approaches aim to find meaningful click patterns from clickstream data and interpret behaviors they mean.[25]

In order to discover the bottom-up approaches, several publications have been published in the last few years, and the most common solutions operate with Feedforward Neural Network or Self Organized Map (SOM), while other papers employ Recurrent Neural Networks (RNNs), like Long Short-Term Memory (LSTM) Networks. In this paper, we used Gated Recurrent Unit (GRU) which is introduced by Cho, et al. in 2014. GRU is a variation on the LSTM (Long Short-Term Memory) and it is a specific RNN architecture designed to model temporal sequences and their long-range dependencies more accurately than conventional RNNs [14].

Although, during the last few years several papers have been published which started to use Deep Learning to predict MOOC outcomes, the main concept has not change. Kim et al. (2018) [15] collected these studies, so far these have shown low accuracies. The fact that accuracy tends to be low can be related to the model's continued reliance on feature engineering to decrease input dimensions which seems to hamper the development of greater and improved Neural Network models [15]. A good example is R. Al-Shabandar's et al. (2017) [7], in which numerous characteristics were obtained from learners' historical data, including how many sessions they took, how often they watched the videos, how many courses they participated in, which was then all fed into a Feedforward Neural Network. Because of the information loss of feature extraction, current dropout or student performance prediction model's accuracy is limited, and Deep Learning methods could not give much better performance than classic Machine Learning methods.

We could think that the RNN network is not robust enough for raw log-line level data, but on the other hand, other fields – like anomaly detection – some experts applied Long Short-Term Memory (LSTM) Networks straight on unprocessed data. Zhang et al. [26] for instance, opted for using clustering techniques for the raw text from numerous log sources so as to create feature sequences fed to an LSTM for hardware and software failure predictions. Du et al. [27] implemented special parsing methods on the unprocessed text of system logs to create sequences for LSTM Denial of Service attack identification [28].

We address this challenge here by using RNNs directly on raw log-line level data, rather than hand-designed feature extracted statistics.

3 Methodology

In terms of algorithm, this work studied the power of Deep Learning in the context of education. The authors' prediction algorithm is based on raw clickstream data in order to provide a prediction of how well students will perform in an online course. We use the students' final assessment quiz responses to define the course performance measure. To do this, we measure the user performance prediction in two ways:

- as a regression problem (0-100%), and
- a multiclass problem (0-9 point). To get labels for the multiclass, we used uniform discretization of continuous data (0-10% = 0, 11-20% = 2, etc.).

In order to compare our research with other solution, we implemented the Baseline solutions, i.e. feature extractors on raw clickstream and used traditional classifiers and regressors on these feature set. We compare the Baseline solution to a GRU-based model on raw log-line level data (GRU).

3.1 Evaluation

To measure the accuracy of the prediction approaches, we build temporal performance prediction models on a weekly basis. The log data observed on the second week were directly added to the dataset on the first week, and similarly for other weeks as well. We are using the data collected until the current week to predict the student's outcome in the very end of the course as shown in Figure 1. We extract the commulated features from the collected data for the Baseline solutions and train a GRU model on the raw data.

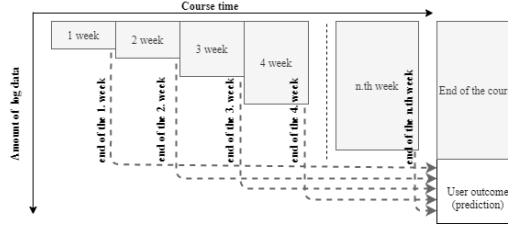


Fig. 1. Formulation of performance prediction problem

After the training step, we evaluate the accuracy of the prediction models on the hold-out set of students. The predictions on this hold out set is compared against the true value (either the actual score or the performance class) of the students' final course performance. Following Willmott and Matsuura (2005) [6], for measuring the accuracy of the solutions, we use accuracy scores (ACC) for classification and root mean squared error (RMSE) for regression. We split the students into four groups randomly and employ a 4-fold cross-validation at each week.

Certainly, one of the obstacles to be overcome when creating predictions in MOOCs is how to deal with sparse data [29]. In Stanford Lagunita's datasets, the majority of learners failed to provide answers to every quiz question in the relevant

MOOC, which in turn, resulted in a sparse set of quiz replies for each individual student. To handle this problem, we only use those users who has filled second quiz after the first week, or filled third quiz after the second week, etc. To avoid under or over learning problem, we moved extreme outliers [30] from datasets.

3.2 Baseline solutions

For the baseline solutions, we extracted commulated features from the raw data (see in Table I.). All of these features are normalized by min-max scaling with the maximum and minimum values of each feature in training dataset.

In this study, three popular Machine Learning algorithms were employed from the gradient boosting framework, and Ridge Regression to train and predict over the cumulated feature set. We used the gradient boosting (XGboost) for classification and Ridge Regression, XGBRegression for regression tasks. Our main goal is not to find the most accurate model, so we did not perform any hyper-parameter tuning.

Table 1. Feature set of Stanford’s course

| Feature | Explanation (feature aggregated on a weekly basis) |
|---------------------|--|
| Lecture view | Number of lecture videos viewed by a student |
| | Number of times a student visits a lecture site |
| | Number of plays, stop, pause, forward, backward |
| | Number of plays, stop, pause, forward, backward |
| | The distance of time between two log events |
| Quiz attempt | Number of quizzes attempted by a student |

3.3 Deep Learning architecture

We propose a Deep Learning method based on raw data, to compare the prediction performance with baseline algorithms.

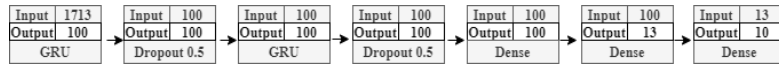


Fig. 2. Architecture overview of the proposed RNN model.

Our Recurrent Neural Network architecture consists of GRU cells as hidden units. In the model training, hidden states learned raw log-line by raw log-line to catch sequential information, which is able to absorb past information. Our GRU model also takes the dropout technique to prevent overfitting in training process. As the experiment result has shown in the result section, the usage of GRU and dropout technique helps us to achieve the training goal. Although there are various suitable algorithms in terms of this prediction, the authors opted for time series/sequence-based neural network predictor, mainly based on its frequent implementation in numerous research fields, thus also student knowledge tracing [29]. Sequence-based neural networks are Recurrent Neural Networks, with feedback connections enclosing several layers of the network. Gated Recurrent Unit (GRU) networks is an example of Recurrent Neural Networks.

GRU is a good choice at solving problems that require learning long-term temporal dependencies. Figure 2 depicts the architecture of our proposed Deep Learning model consisting of nine layers.

The input layer of our deep network uses a flat feature structure (one hot encoded 3-dimensional data), as we can see in figure 3. The rows in our 3D dataset contain user-generated log data. These rows are representing a user-generated 2D sequence of actions, which set up in chronological order. In this 2D sequence, a line is a 1714 length vector, which represents one of the 1713 possible actions which illustrated with a One Hot Encoded vector, along with a single feature which is the time elapsed since the last action. An action encodes either the type of action (e.g. “video stop”) or the item the was accessed. For example, when the student opens “Lecture1 Part3” which is a webpage containing a lecture video we log this event. Next, when the student plays the video, we add a new action to the sequence, but we only store its action type “video play”. While this condensation of data is necessary for keeping the input space at tractable size, we expect from the RNN, that it can learn the representation of items in its hidden states.

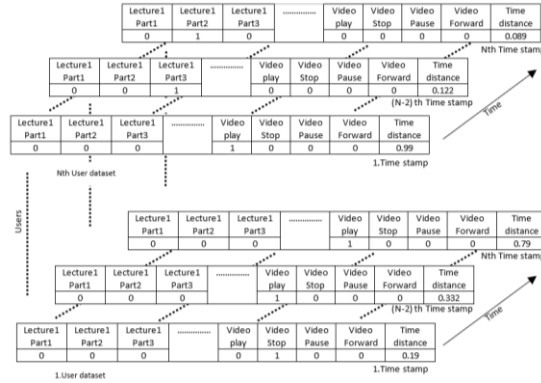


Fig. 3. Fig. 1. Formulation of 3-dimensional data

RNN is likely to quickly overfit a training dataset. To reduce the chance of overfitting, we used dropout layers which offers a very computationally cheap and remarkably effective regularization method to reduce overfitting and improve generalization error in our model.

Following the GRU and dropout layers we applied fully connected (dense) hidden layers with different number of neurons. We use the same network architecture for the regression and classification tasks besides the output layer, At the regression task, we use the result from the single output neuron without any transformation. At the multi-class classification task, the output layer consists of a vector that contains values for each class along with a SoftMax activation function.

4 Experimental Setup

In our experiments, we used the activity log dataset of the Stanford Lagunita’s MOOC Computer Science 101 from the summer of 2014. The MOOC ran for six weeks, with video lectures, optional homework assignments, discussion forums, and quizzes. The original main dataset contains 39.6 million action items from around 142,395 students, where each action represents accessing a particular event in the course, like video view, assignment view, problem view. From the 142,395 students, 28,368 were active (went through almost every curriculum) and 12,015 completed enough assignments and scored high enough on the exams to be considered “certified” by the instructors of the course. The certified students accounted for 17.79 million of the original 39.6 million actions, with an average of 1,135 action items per certified student. In our research the set of 12,015 students were used.

To assess our model’s ability to spin up rapidly and predict final outcome, we limit our scope to first 5 weeks, and we make a prediction after every week with the baseline and GRU models. At each week, we compared baseline and Deep Learning solutions for both a regression and classification tasks. In the regression task, the goal was to predict student performance over a range of 0-100%, while in the multiclass classification problem we targeted seven classes of student performance. In each experiment, we employed a 4-fold cross-validation over students. In the GRU, our learning rate is fixed to 0.001; for training we used the ADAMAX optimizer; we didn’t use padding, instead we create variable sequence length for all user with fit generator function; our model used two GRUs layer with 100 hidden units; two Dropout layers (ranged 50%), two Dense layers with 100 hidden units each, and one Dense with 13 hidden unit.

The experiments were implemented in python by using Keras [1] Google’s TensorFlow [2], Scikit learn [3] and XGBoost [4] package.

5 Empirical Results

We ran our models at the end of each week, i.e. after a quiz, and based on that information, we predicted the final completion of the course. For example, *week1* represents all collected log data from the start of the course until the end of the first week, and *week2* represents the collected data until the end of the second week. The results of the proposed GRU method and the baseline methods on Computer Science 101 dataset are shown in Table 2 respectively.

TABLE 2. Results Of Gru And By Baseline Methods

| | RMSE | | | MAE | | | Accuracy | |
|-------|---------------|--------------|----------------|---------------|--------------|----------------|----------------|------------------|
| | <i>XGBreg</i> | <i>Ridge</i> | <i>GRU-reg</i> | <i>XGBreg</i> | <i>Ridge</i> | <i>GRU-reg</i> | <i>XGBoost</i> | <i>GRU-class</i> |
| week1 | 16.010 | 18.451 | 10.001 | 11.706 | 12.854 | 7.743 | 0.361 | 0.496 |
| week2 | 16.217 | 19.173 | 9.831 | 11.784 | 13.564 | 7.451 | 0.378 | 0.486 |
| week3 | 15.730 | 20.779 | 9.378 | 11.251 | 13.101 | 7.117 | 0.39 | 0.545 |
| week4 | 15.346 | 29.207 | 8.746 | 10.202 | 15.378 | 6.096 | 0.405 | 0.559 |
| week5 | 15.265 | 25.355 | 8.653 | 10.585 | 15.280 | 6.568 | 0.405 | 0.551 |

The results of the two experiments in Table 2 shows that the GRU model is generally better than the XGBoost-regression and XGBoost and both RNN based models increase their prediction quality week by week. This shows that RNNs with raw datasets have more sensitivity to “catch” patterns than XGBoost or XGBoost-regression. This was not surprising, as standard machine learning methods without deep feature investigation cannot make evaluable results. In addition to basic transformations (sum, avg, normalization), we also used other methods to increase the performance of the XGBoost model, as discussed in Chapter 3, but apparently this solution was not sufficient. Remarkably, for all week in the proposed method outperforms the best baseline by 15% of ACC or 30 of RMSE. As expected, performance starts out poor (low ACC or high RMSE) but steadily improves as more data are fed into the model. Predictions continue to improve until approximately week 3, after which performance levels out. After 3 weeks of observations, our model achieves an accuracy of 54%, significantly better than the 39% baseline of predicting the class role.

Our empirical results demonstrate the feasibility of using Recurrent Neural Networks on raw log-line level dataset to predict MOOC students’ performance. We do expect, however, that a more extensive search for the optimal choices of number of units and hidden layers (through e.g., hyper-parameter tuning, embedding layer) will improve our prediction quality further.

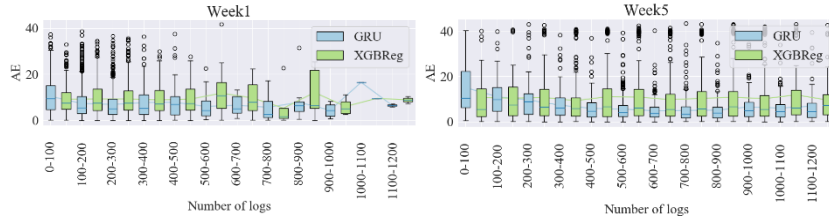


Fig. 4. Boxplots of average error (AE) achieved by GRU-reg and XGBReg in the function of student’s log sequence length

To better understand our results, we performed other examination (see in Fig. 4.), where the relation between the number of log-data and the absolute error (AE) of the model is plotted. In this process we calculated AE for every user falling in a particular bin of log-data sequence length and made a boxplot form this data to compare two models’ outputs. In the first two weeks, the GRU network provides better results than the traditional approach on any log size. In weeks 3,4,5 traditional approaches outperform RNNs at short log sequences and GRU is only superior when it has got more data of longer sequences that provide extra information.

Other problem is length of the sequence, which is not same long for all users. For example, some users finish 6-week course very fast and short way (exp. Doesn’t check additional information), while other users look into everything and spend more time checking the details. These results also show that there is a significant correlation between the length of the sequences and the prediction quality of the RNN.

RNN model do temporal processing and learn from sequences, e.g., perform sequence recognition or temporal association, which is impossible to solve with

regression. On the other hand, XGBreg is more sensitive to overfit. To avoid XGBoost-regression disadvantages, we experimented with various regression models, but they provide almost the same or worse result (Ridge regressor was among the best one, see its results in Table 2).

This fact is also supported by our prediction density distribution graph of real and predicted course outcomes. (Fig 5). The diagram shows that XGBReg was not able to find any useful relationship among its features. On the other hand, GRUreg was capable to identify different class-like regions in its prediction space. The results are not exhaustive, but they do encourage us to explore further.

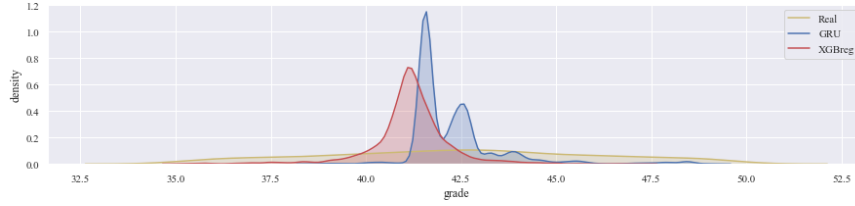


Fig. 5. The distribution of Student's final outcomes - The test results of the proposed GRU method and the baseline (XGBRegression) methods on Computer Science 101 dataset at fourth week (n = 2159)

6 Conclusions

In this study, we propose a Recurrent Neural Network for solving outcome performance prediction problem in online learning platform. The main task of this paper is to build prediction model which could use raw datasets, and get same or better results than regular prediction models. The key advantage of our model is that, there is no manual feature engineering is needed, because it could be automatically extracted from the raw log-line level records. In this way, this approach could save a lot of time and human force, and ignore the possible inconsistency introduced by the hand-made process. Experimental results on Stanford Lagunita's dataset show that the expected model can achieve significantly better than the baseline models. The results for our model are sufficient to demonstrate the feasibility of using Recurrent Neural Networks. Additionally, architectural insights can be gleaned from the RNN applied, which may in the future lead to designing more effective models. The methodology in this study aimed to predict the outcome performance of student participations in MOOCs. In table 2, we see that our GRU-Reg and GRU-Class algorithms are especially useful for predicting the performance of students who answer quizzes. Our neural networks give much better results than regular solutions. On the other hand, we compare a high-end solution with poor predictors, an important step for the feature work would be to implement a well-designed baseline model which could be enough powerful for the "fair play". In this paper our main goal, is not to find the most accurate model, but in the next possible interventions could be to improve these parameters from 55% to 80%. In the future, we still need to take some optimization techniques to adjust hyper parameters of the proposed RNN model, which is helpful to improve the accuracy. There are several ways that we plan to extend this ongoing work. First, we would like to use a hyper-parameter

tuning to find the best number of units and layers in GRU network. We also plan to address the role imbalance by resampling and improve generalization, and layer normalization. Finally, we plan to use embedding layer which could give us an information what is happened inside of the neural network, this work also includes evaluating the system on different datasets which will provide by Stanford Lagunita.

Acknowledgment We are grateful to the Leland Stanford Junior University, The Office of the Vice Provost for Online Learning, Stanford Lagunita for providing access to their MOOC log dataset. This work was supported by the National Research, Development and Innovation Office of Hungary through the Artificial Intelligence National Excellence Program (grant no.: 2018-1.2.1-NKP-2018-00008) and TUDFO/47138-1/2019-ITM of the Ministry for Innovation and Technology, Hungary. We are also grateful for Tamás Hegedűs and Tamara Sehovac who conducted data preprocessing and technical preparation of the experiments and whose work was funded from the "Integrated program for training new generation of scientists in the fields of computer science", no EFOP-3.6.3-VEKOP-16-2017-0002, supported by the European Union and co-funded by the European Social Fund.

References

1. Chollet, F., et al.: Keras (2015)
2. Martín A., et al.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015)
3. Pedregosa, et al.: Scikit-learn: Machine Learning in Python, JMLR 12 (2011) 2825–2830
4. Chen, T., Guestrin, C.: XGBoost: A Scalable Tree Boosting System, Proceedings of the 22nd ACM SIGKDD International Conference on KDD '16 (2016) 785–794
5. Gardner, J., Brooks, C.: Student success prediction in MOOCs, User Modeling and User-Adapted Interaction, June 2018, Volume 28, Issue 2 (2018) 127–203
6. Cort, J. W., Kenji, M.: Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance, Climate Research, vol 30 (2005) 79–82
7. Al-Shabandar, R. et al.: Machine Learning Approaches to Predict Learning Outcomes in Massive Open Online Courses, 2017 International Joint Conference on Neural Networks (IJCNN) (2017) 713–720
8. Fei, M., Yeung, D. Y.: Temporal models for predicting student dropout in massive open online courses, Proceedings of the 2015 IEEE International Conference on Data Mining Workshop (2015) 256–263
9. Whitehill, J., Williams, J., Lopez, G., Coleman, C., Reich, J.: Beyond prediction: Toward automatic intervention to reduce MOOC student stopout. In Educational Data Mining (2015)
10. Xing, W., Du, D.: Dropout Prediction in MOOCs: Using Deep Learning for Personalized Intervention, Journal of Educational Computing Research (2018)
11. Jordan, M.I.: Attractor dynamics and parallelism in a connectionist sequential machine, in Artificial Neural Networks. IEEE Press (1990) 112–127
12. Kloft, M., Stiehler, F., Zheng, Z., Pinkwart, N.: Predicting MOOC Dropout over Weeks Using Machine Learning Methods, Proc. 2014 Conf. Empir. Methods Nat. Lang. Process (2014) 60–65

13. Yang, T.S., Brinton, C. G., Chiang, M.: Behavior-Based Grade Prediction for MOOCs Via Time Series Neural Networks, *Ieee journal of selected topics in signal processing*, vol. 11, no. 5 (2017a) 716–728
14. Liu, Z., Xiong, F., Zou, K., Wang, H.: Predicting Learning Status in MOOCs using LSTM, *ACM TURC '19 Proceedings of the ACM Turing Celebration Conference – China* Article No. 74 (2019) 74–81
15. Kim, B., Vizitei, E., Ganapathi, V.: Gritnet: Student performance prediction with deep learning, *Educational datamining* (2018)
16. Whitehill, J., Mohan, K., Seaton, D., Rosen, Y., Tingley, D.: MOOC Dropout Prediction: How to Measure Accuracy? *ACM Conference on Learning* (2017a) 161–164
17. Fiaidhi, J.: The next step for learning analytics, *IT Professional*, vol. 16 (2014) 4–8
18. Baker, R.S.J.D. Siemens, G.: Educational Data Mining and Learning Analytics, in *Cambridge Handbook of the Learning Sciences* (2014)
19. Tang, S., Peterson, J. C., Pardos, Z. A.: Predictive Modelling of Student Behavior Using Granular Large-Scale Action Data, In book: *Handbook of Learning Analytic*, Publisher: Society for Learning Analytics (2017)
20. Gardner, J., Brooks, C., Andres-Bray, M.L.J, Baker, R.: Replicating MOOC predictive models at scale ,*L@S '18 Proceedings of the Fifth Annual ACM Conference on Learning at Scale* Article No. 1 (2018) 25–37
21. Whitehill, J., Mohan, K., Seaton, D., Rosen, Y., Tingley, D.: Delving deeper into MOOC student dropout prediction (2017b)
22. Gavai, G., Sricharan K., Gunning, D., Hanley, J., Singhal, M., Rolleston, R.: Supervised and unsupervised methods to detect insider threat from enterprise social and online activity data, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* 6(4) (2015) 47–63
23. He, J., Bailey, J., Rubinstein, B.I.P.: Identifying At-Risk Students in Massive Open Online Courses, *Proc. 29th AAAI Conf. Artif. Intell.* (2015) 1749–1755
24. Kizilcec, R.F., Piech, C., Schneider, E.: Deconstructing Disengagement : Analyzing Learner Subpopulations in Massive Open Online Courses, *Lak '13* (2013)
25. Jo Y., Maki, K., Tomar, G.: Time Series Analysis of Clickstream Logs from Online Courses, *Language Technologies Institute, Carnegie* (2018)
26. Zhang, K, Xu J., Min R. M., Jiang G., Pelechrinis K., Zhang H.: Automated IT system failure prediction: A deep learning approach. In *Big Data (Big Data)*, 2016 IEEE International Conference on. IEEE (2016) 1291–1300
27. Du, M., Li, F., Zheng, G., Srikumar, V.: DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning. In *Proceedings* (2017)
28. Brown, A., Tuor, A., Hutchinson, B., Nichols, N.: Recurrent Neural Network Attention Mechanisms for Interpretable System Log Anomaly Detection, *First Workshop On Machine Learning for Computer Systems, ACM HPDC* (2018)
29. Yang, T-Y., Brinton, C.G., Joe-Wong, C.: Behavior-Based Grade Prediction for MOOCs via Time Series Neural Networks, *IEEE Journal of Selected Topics in Signal Processing* (Volume: 11 , Issue: 5 , Aug. 2017) (2017b) 716–728
30. Tukey, J.W.: *Exploratory Data Analysis*, Addison-Wesley (1977)
31. Mourdi, Y., Sadgal, M., El Kabtane, H., Berrada Fathi, W.: A machine learning-based methodology to predict learners' dropout, success or failure in MOOCs, *International Journal of Web Information Systems*, Vol. 15 No. 5 (2018) 489–509
32. Pigeau, A., Aubert, O., Prié, Y.: Success prediction in MOOCs A case study, *Educational Data Mining 2019* (2019)