Personalized Student Performance Prediction Using Multivariate Long Short-Term Memory

Tran Thanh Dien(✉), Pham Huu Phuoc, Nguyen Thanh-Hai, and Nguyen Thai-Nghe(✉)

College of Information and Communication Technology, Can Tho University, Can Tho, Vietnam
{thanhdien,nthai.cit}@ctu.edu.vn,
phuocB1605301@student.ctu.edu.vn, ntnghe@cit.ctu.edu.vn

Abstract. In the age of knowledge economy, with deep international integration in all fields, the requirement for competitive human resources becomes increasingly fierce, deciding the success or failure of a country. The competitiveness of human resources depends much on the training process of the education system, notably the higher education level. Therefore, managing and monitoring student learning results are essential for lecturers, particularly the school in general. Early forecasting of students' learning results is expected to help students choose the suitable modules or courses for their competencies, allowing leaders, administrators, and lecturers in universities and institutes to identify students who need more support to complete their studies successfully. In addition, it contributes to reducing academic warnings or expulsion or suspension due to poor academic performance. Also, this saves time and costs for students, families, schools, and society. This article proposes an approach to enhance student learning performance prediction by applying some deep learning techniques to exploit databases in student management systems at universities in a personalized way. More specifically, we consider personalized training, which means all mark entries of each student are trained separately and apply that trained model to predict scores of courses for themselves. The collected data is analyzed, pre-processed, designed, and prepared with a Long Short-Term Memory network with multiple input variables. Experimental results reveal that the proposed method's average performance outperforms the method that trains whole datasets with an RMSE of 0.461 with a Multivariate Long Short-Term Memory network.

Keywords: Personalized performance · Multivariate Long Short-Term Memory · Multilayer Perceptron · Support Vector Regression

1 Introduction

Over the past years, the enrolment work has been increasingly difficult, but the situation of students in institutions and schools being warned of academic studies or forced to drop out of schools still tends to increase.

One of the main reasons students' poor learning results is that they have not chosen suitable modules for their abilities. The early detection of students who are likely to be

T. K. Dang et al. (Eds.): FDSE 2021, CCIS 1500, pp. 238–247, 2021.
https://doi.org/10.1007/978-981-16-8062-5_16

forced out of school is expected to help students plan their studies accordingly, which is an essential need of the school today. Therefore, many researchers have focused on forecasting student results as an essential research topic in educational data mining.

The work focuses on building models to predict students' learning results using MLSTM (Multivariate Long Short-Term Memory), LSTM (Long Short-Term Memory), MLP (Multilayer Perceptron), and SVR (Support Vector Regression). We also compare and evaluate the effectiveness and suitability of the proposed method with experimental data of various universities. There is no absolute scale to measure knowledge, but grades are an essential indicator of student learning. When these scores are combined with other context indicators in sequence, a multi-attribute time series is formed. We arranged the records chronologically from each student's first to the last semester with the actual scores observed. Additional context attributes include GPA previous semester, cumulative GPA up to the previous semester, count only accumulated up to the previous semester, and course scores form a series of data over time-space has four attributes. The problem using the results of the earlier modules is forecasted for the following modules.

2 Related Work

Academic forecasting is a topic that has received much attention from data scientists. The methods used for forecasting also vary from classical machine learning methods to modern deep learning methods [1–4]. Bharadwaj and Pal [5] used students' previous semester grades, class test scores, workshop performance, task performance, class participation, and laboratory work to predict students' grades in the final semester. Galit et al. [6] presented a case study that used student data to analyze their learning behavior to predict results and warn students at risk before the end of the exam. According to Thai-Nghe et al. [7], forecasting student achievement is essential in educational data mining. Students' knowledge can be improved and accumulated over time. Using a tensor factorization (TF), the matrix decomposition technique has been proposed to predict student learning outcomes from this idea. With this approach, the author's team can personalize the forecast for a particular student. Experimental results on two large data sets show that incorporating prediction techniques into the matrix decomposition process is effective and promising approach.

The work in [8] performed experiments conducted on a final-year university module student cohort of 23. Individual student data are limited to lecture/tutorial attendance, virtual learning environment accesses, and intermediate assessments. [9] investigated blended learning by implementing a student-centered teaching method based on the flipped classroom and miniature private online course. Al-Radaideh et al. [10] applied the decision tree model to predict students' final scores who took C++ courses at Yarmouk University, Jordan, in 2005. The three different classification methods, ID3, C4.5, and NaiveBayes, were used. Their results indicated that the decision tree model was better than predicted by other models. Authors [11] built a predictive model of student learning outcomes using a multi-layer neural network model to explore valuable information from Can Tho university's student management system data. The forecast results show better accuracy than the ItemKNN, MatrixFactorization, ItemAverage, and UserAverage methods. Some recent works also reveal exciting results in student performance prediction [12–18]. Open-source libraries (Keras [19], Sklearn [20], Theano - LISA [21]) of the

Python programming language in machine learning, deep learning is highly developed and widely used in recent times.

3 Method

This paper proposes a novel approach which can improve predictive model from conventional neural networks - a Long Short-Term Memory network with many properties as network input for training. There can be no absolute scale to measure knowledge, but grades are an essential indicator of student learning. When these scores are combined with other context indicators in sequence, a multi-attribute time series is formed.

3.1 Datasets for the Experiments

The data used for the experiment includes two data sets. The first volume is extracted from the student management system of Can Tho University, Vietnam (CTU dataset) from 2017 to 2019. The set includes more than 1 million observations of 94,087 students in some faculties and institutes with 4,836 different courses. The other group is extracted from the student management system of Vinh Long University, Vietnam (VLU dataset) from 2014 to 2020. The set includes 280,493 observations with 6,255 students from 14 disciplines, 569 other courses.

3.2 Data Pre-processing

Collected data needs to be processed to suit the training and testing modeling process. The basic steps are as follows:

- Step 1: Sort the data by the student and chronologically from the first semester to the last semester.
- Step 2: Remove unused data attributes for the model and retain the required features. Eliminate students with too few observations. We have four new data sets from 2 original data sets with four different filters to diversify training and testing data. Details are described in Table 1.

Table 1. Description of data used in the experiments

Dataset	#Sample	Description
CTU_01	515,610	Extract score entries of students who have more than ten mark entries
VLU_01	268,134	
CTU_02	41,367	Extract score entries of students who have more than 20 mark entries
VLU_02	230,045	

We add more fields to match the training and testing model:

– StudentID: Unique identifier of the student (this column is only used to identify the data rows of each student, not used in the forecast model).
– CGPA: GPA accumulated up to the previous semester.
– CGPA-PreSemester: Cumulative GPA of the previous semester.
– Credits-earned: Total calculated only accumulated to the previous semester.
– Mark: points of the course (or subject) on a 4-point scale (from 0 to 4).
– Step 3: Removing noise and pre-processing data for students' scores (exemption scores, scores not completed courses, course withdrawal points, students who registered but did not participate in study (null/NaN)). If the mark record is in the student's first semester, we set 0 to the GPA of the semester.

3.3 Learning Models for Student Performance

Building MLSTM Prediction Model: MLSTM model built after the standardization of the input data. Input data transformation is done through the following steps:

- Step 1: Read a student's data into memory. The column of the mark is the column to be forecasted (as illustrated in Fig. 1).

	StudentID	CGPA	CGPA-PreSemester	Credits-earned	Mark
0	B1206286	0.00	0.00	0.00	3.16
1	B1206286	0.00	0.00	0.00	3.52
2	B1206286	0.00	0.00	0.00	3.52
3	B1206286	0.00	0.00	0.00	1.88
4	B1206286	3.02	3.02	14.0	2.76
5	B1206286	3.02	3.02	14.0	3.84
6	B1206286	3.02	3.02	14.0	3.52
7	B1206286	3.02	3.02	14.0	2.80

Fig. 1. Illustration of student records

- Step 2: Delete the StudentID column after data of each student is read into memory, then get the array values of the records (Fig. 2)

```
[ 0.   ,  0.   ,   0.  ,  3.16 ],
[ 0.   ,  0.   ,   0.  ,  3.52 ],
[ 0.   ,  0.   ,   0.  ,  3.52 ],
[ 0.   ,  0.   ,   0.  ,  1.88 ],
[ 3.02 ,  3.02 ,  14.0 ,  2.76 ],
[ 3.02 ,  3.02 ,  14.0 ,  3.84 ],
[ 3.02 ,  3.02 ,  14.0 ,  3.52 ],
[ 3.02 ,  3.02 ,  14.0 ,  2.80 ]]
```

Fig. 2. Data after removing unnecessary fields

- Step 3: Large Input values can slow down the learning and convergence of the network. Besides, it can affect model training time and also reduces model quality in some cases. Therefore, scaling values (e.g., −1 to 1) are necessary before being fetched into the network. It is also an advantage for *Tanh*.
- Step 4: Transform data into multivariate input data, then using values at the time (*t-1*) predict time *t*. In this example, the data column to be indicated is *var4* (Fig. 3).

	var1(t-1)	var2(t-1)	var3(t-1)	var4(t-1)	var4(t)
1	-1.	-1.	-1.	0.306122	0.673469
2	-1.	-1.	-1.	0.673469	0.673469
3	-1.	-1.	-1.	0.673469	-1.000000
4	-1.	-1.	-1.	-1.000000	-0.102041
5	1.	1.	1.	-0.102041	1.000000
6	1.	1.	1.	1.000000	0.673469
7	1.	1.	1.	0.673469	-0.061224

Fig. 3. Transformed data

After the data transformation steps, we build the MLSTM model with the support of the Keras library.

The LSTM network architecture consists of Input data (sequences of time steps), an LSTM layer with 50 nodes, and a hidden layer (Dense layer). In addition, there is one node for the prediction result, a value in the range of 0 to 4 (Fig. 4).

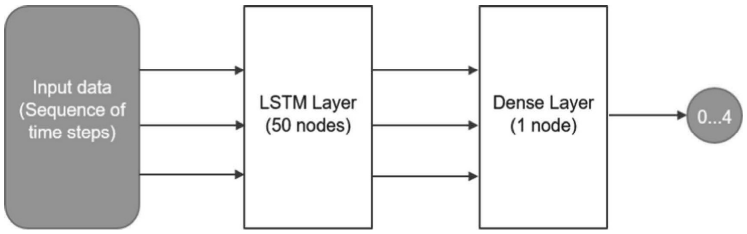

Fig. 4. LSTM network architecture

To reduce overfitting, we use the Early Stopping technique when training. On five consecutive epochs, if the error does not change by more than 0.01, we stop training. Thus, if overfitting does not occur, training will run at a maximum of 100 epochs. In addition, this technique also helps us to reduce the training time of the model significantly.

Building LSTM Prediction Model: The difference between the traditional LSTM model and MLSTM is the input data of the network. The LSTM network uses a value at a time *(t-1)* to predict the value at time *t*. The LSTM network input univariate data is shown in Fig. 5. Similar to the MLSTM model, the univariate LSTM network architecture also uses sequential chronological data as input. The LSTM layer has 50 nodes, and a hidden layer has one node. The results of the prediction are values between 0 and 4.

	Var1(t-1)	Var1(t)
1	0.306122	0.673469
2	0.673469	0.673469
3	0.673469	-1.000000
4	-1.000000	-0.102041
5	-0.102041	1.000000
6	1.000000	0.673469
7	0.673469	-0.061224

Fig. 5. Univariate data

Building MLP Prediction Model: Using the Keras library, we build a detailed architectural MLP model consisting of input data and five hidden layers. the first hidden layer has nine nodes using the rectified linear unit (ReLU). The second and third hidden layers have 27 nodes, using the activation function of sigmoid. The fourth hidden layer has nine nodes, using ReLU. Finally, the Fifth hidden layer has one node for an output value range from 0 to 4. We use the early stopping technique in training. The architecture of the MLP network is shown in Fig. 6.

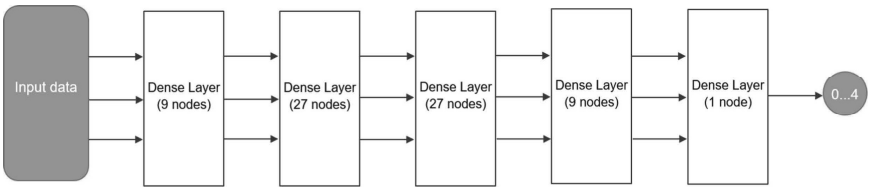

Fig. 6. The architecture of the MLP network

Building Support Vector Regression (SVR) Prediction Model: With the help of the scikit-learn machine learning library [20], the SVR model uses the radial basis function kernel (RBF). This kernel is suitable for multi-variable and nonlinear data.

4 Evaluation of Results

4.1 Evaluation Method

We use the standard error calculation method to evaluate the model, the square root of the mean square error (RMSE). The formula for calculating the RMSE error is as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - F_t)^2}$$

Where A_t is the t-th sample real value and F_t is the t-th sample prediction value, n is the number of samples used for evaluation.

4.2 Experimental Results

The MLSTM and LSTM methods are trained on each student. For each student, we separate the data into two parts, 90% of them for the training stage and 10% remaining for the testing stage. After training and testing, the RMSE is an average error of all students.

The MLP and SVR methods train all students. So for each student, we get 90% of the first records and put them into the same training data set and 10% remaining records put into the same testing data set.

The details of the data distribution in the training and testing data set of CTU_01, CTU_02, VLU_01, and VLU_02 are illustrated as Fig. 7, Fig. 8, Fig. 9, and Fig. 10, respectively. For example, in the CTU_01 data set, the distribution of marks focuses on marks greater than or equal to 3, especially in B and B+. On the other hand, the distribution mark A in the testing set is more than one in the training set. However, the difference is not too significant.

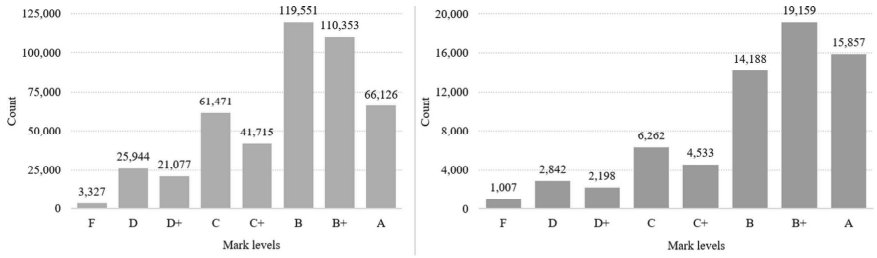

Fig. 7. Mark levels data distribution of dataset CTU_01 (on the left, we present the data distribution of training set, the other chart is for the testing set)

Similar to the CTU_01 dataset, the difference between training and testing sets in the CTU_02 data set is not too significant. However, mark A in the testing set is somewhat more than the training set. Therefore, it can be that the student's marks significantly improve academic performance in the following semesters compared to the first semester.

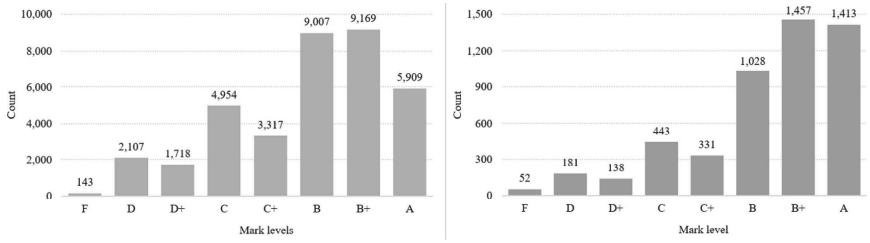

Fig. 8. Mark levels Data distribution of dataset CTU_02

Figure 9 shows the distribution of the training and testing data of the VLU_01 set. Again, we can see that the distribution of the training and testing set is very similar.

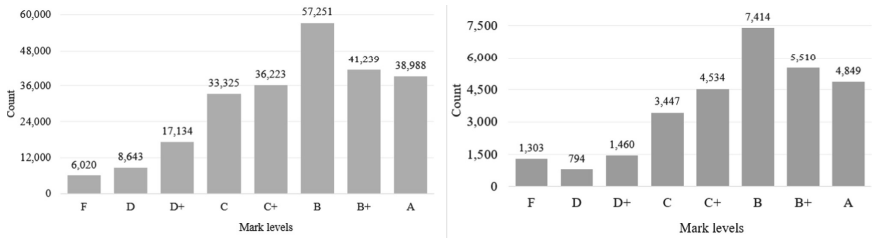

Fig. 9. Mark levels data distribution of dataset VLU_01

Figure 10 shows the distribution of marks in the VLU_02 data set. Again, the distribution in patterns of the two groups has no significant difference. Thus, two data sets of VLU students show that students' learning results have a slight variation from the previous semesters to the following semesters. One note issue is that the F marks have a subsequence trend in the following semesters of VLU students. Although the rate of increase is not too large, it partly reflects the students' difficulty in specialized courses.

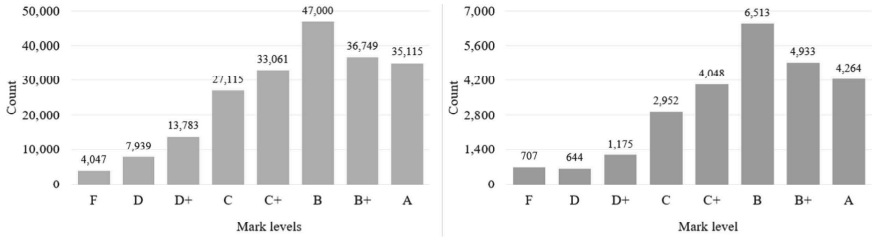

Fig. 10. Mark levels Data distribution of dataset VLU_02

In general, the percentage of marks in training and testing sets in the data sets is quite similar. Therefore, it can be a good effect on the quality of the predictive models.

In this study, we evaluated the models by comparing the RMSE error between methods. Due to the randomness of the algorithm, in MLP architecture, we examine the model ten times, then calculate the average RMSE error.

Table 2 shows the experimental results of the models. The results of the RMSE error comparison show that LSTM and MLSTM models achieve better results than other methods on the same data set. As we can see, the LSTM network works quite well on data of sequences of time steps in both the univariate and multivariate data problems. Moreover, when there are more records of marks, the MLSTM model can improve prediction results. Consequently, the MLSTM model can be more suitable for big data. Moreover, multi-time step data is also suitable conditions for the MLSTM model.

Table 2. Prediction Performance comparison among the considered methods

Dataset	RMSE			
	MLP	SVR	LSTM	MLSTM
CTU_01	0.536	0.525	0.505	**0.502**
VLU_01	0.564	0.558	0.511	**0.502**
CTU_02	0.526	0.525	0.513	**0.501**
VLU_02	0.532	0.512	0.468	**0.461**

5 Conclusion

This work introduces a novel method to predict student performance based on the data extracted from the student management system at universities. The results show that the proposed method used for separate students can generate more accurate predictions than the model trained by whole students (a model for a complete dataset). The proposed model was evaluated on two datasets. However, the method can be applied to other datasets at other universities in the same way.

By tracking students' grades and receiving early warnings, the system can help students organize their study plans, and teachers can better monitor and support students. It also improves the quality of education, reduces training time, helps save costs, and develops society. In the future, we continue to improve the quality of the model in several different ways, such as collecting more data attributes that consist of other time factors, including training grades, learning behaviours, etc.

References

1. Ma, Y., Cui, C., Yu, J., Guo, J., Yang, G., Yin, Y.: Multi-task MIML learning for pre-course student performance prediction. Front. Comp. Sci. **14**(5), 1 (2019). https://doi.org/10.1007/s11704-019-9062-8
2. Dien, T., Hoai, S., Thanh-Hai, N., Thai-Nghe, N.: Deep Learning with data transformation and factor analysis for student performance prediction. Int. J. Adv. Comput. Sci. Appl. **11**(8), 711–721 (2020). https://doi.org/10.14569/ijacsa.2020.0110886
3. Ünal, F.: Data mining for student performance prediction in education. data mining – methods. Appl. Syst. (2021). https://doi.org/10.5772/intechopen.91449.
4. Minn, S.: BKT-LSTM: efficient Student Modeling for knowledge tracing and student performance prediction. arXiv.org (2021). https://arxiv.org/abs/2012.12218
5. Yadav, S., Pal, S.: Data mining: a prediction for performance improvement of engineering students using classification. arXiv.org (2021). https://arxiv.org/abs/1203.3832
6. Prasad, G.N.R., Babu, D.A.V.: Mining previous marks data to predict students performance in their final year examinations. Int. J. Eng. Res. **2**(2), 1–4 (2013)
7. Nghe, N., Schmidt-Thieme, L.: Factorization forecasting approach for user modeling. J. Comput. Sci. Cybern. **31**(2) (2015). https://doi.org/10.15625/1813-9663/31/2/5860
8. Wakelam, E., Jefferies, A., Davey, N., Sun, Y.: The potential for student performance prediction in small cohorts with minimal available attributes. Br. J. Edu. Technol. **51**(2), 347–370 (2019). https://doi.org/10.1111/bjet.12836

9. Xu, Z., Yuan, H., Liu, Q.: Student performance prediction based on blended learning. IEEE Trans. Educ. **64**(1), 66–73 (2021). https://doi.org/10.1109/te.2020.3008751
10. Al-Radaideh, Q., Al-Shawakfa, E., Al-Najjar, M.: Mining student data using decision trees. In: The International Arab Conference on Information Technology, pp. 1–5. Yarmouk University, Jordan (2006)
11. Sang, L., Dien, T., Nghe, N., Hai, N.: Predicting student's performance through deep learning using a multi-layer perceptron (in Vietnamese). Can Tho Univ. J. Sci. **56**(3), 20–28 (2020). https://doi.org/10.22144/ctu.jvn.2020.049
12. Wei, H., Li, H., Xia, M., Wang, Y., Qu, H.: Predicting student performance in interactive online question pools using mouse interaction features. Proc. Tenth Int. Conf. Learn. Anal. Knowl. (2020). https://doi.org/10.1145/3375462.3375521
13. Kőrösi, G., Farkas, R.: MOOC performance prediction by deep learning from raw clickstream data. In: Singh, M., Gupta, P.K., Tyagi, V., Flusser, J., Ören, T., Valentino, G. (eds.) ICACDS 2020. CCIS, vol. 1244, pp. 474–485. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-6634-9_43
14. Khan, A., Ghosh, S.K.: Student performance analysis and prediction in classroom learning: a review of educational data mining studies. Educ. Inf. Technol. **26**(1), 205–240 (2020). https://doi.org/10.1007/s10639-020-10230-3
15. Yulianto, L.D., Triayudi, A. Sholihati, I.D.: Implementation educational data mining for analysis of student performance prediction with comparison of K-nearest neighbor data mining method and decision tree C4.5. J. Mantik **4**(1), 441–451 (2020)
16. Mengying, L., Xiaodong, W., Shulan, R., Kun, Z., Qi, L.: Student performance prediction model based on two-way attention mechanism. J. Comput. Res. Dev. **57**(8), 1729–1740 (2020). https://doi.org/10.7544/issn1000-1239.2020.20200181
17. Liu, D., Dai, H., Zhang, Y., Li, Q., Zhang, C.: Deep knowledge tracking based on attention mechanism for student performance prediction. In: 2020 IEEE 2nd International Conference on Computer Science and Educational Informatization (CSEI) (2020). https://doi.org/10.1109/csei50228.2020.9142472
18. Zakaria, A., Selamat, A., Fujita, H., Krejcar, O.: The best ensemble learner of bagged tree algorithm for student performance prediction. Knowl. Innov. Intell. Softw. Methodol. Tools Tech. (2020). https://doi.org/10.3233/faia200552
19. Ketkar, N.: Deep learning with python: a hands-on introduction. Apress (2017). https://doi.org/10.1007/978-1-4842-2766-4
20. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O.: Scikit-learn: machine learning in python. arXiv.org (2021). https://arxiv.org/abs/1201.0490
21. Robson, T., Cornish, N., Liu, C.: The construction and use of LISA sensitivity curves. Class. Quantum Gravity**36**(10), 105011 (2019). https://doi.org/10.1088/1361-6382/ab1101