Integrating Deep Learning Architecture into Matrix Factorization for Student Performance Prediction

Thanh-Nhan Huynh-Ly[1,2,3(✉)], Huy-Thap Le[1], and Nguyen Thai-Nghe[4]

[1] Department of Information Technology, Lac Hong University, Bien Hoa, Dong Nai Province, Vietnam
[2] Department of Information Technology, An Giang University, Long Xuyen, An Giang Province, Vietnam
hltnhan@agu.edu.vn
[3] Vietnam National University, Ho Chi Minh City, Vietnam
[4] College of Information and Communication Technology, Can Tho University, Can Tho City, Vietnam
ntnghe@cit.ctu.edu.vn

Abstract. In universities using the academic credit system, choosing elective courses is a crucial task that significantly affects student performance. Because of poor performances, numerous students have been receiving formal warnings and expulsions from universities. Certainly, a good study plan from course recommendation methods plays an important role in obtaining a good study performance. In addition, early warnings that release on challenging courses enable students to prepare better for such courses. Predicting student learning performance is a vital factor in the courses recommendation system and is an essential task of an academic advisor. Many research methods solved this problem with diverse approaches such as association rules, deep learning, and recommender systems (RS). It recently built the courses recommendation system, which is used for personalized recommendation, especially the matrix factorization (MF) technique; But, the prediction accuracy of the MF still need to be improved. So, many studies try to integrate more information (e.g., social networks, course relationships) into the model. Besides, deep learning addresses the student performance prediction, which currently is state of the art, but it usually is general rules (not a personalized prediction). Indeed, deep learning and matrix factorization have advantages and disadvantages, so they need to compound together to get better. This paper proposes an approach to predict student performance that utilizes the deep learning architecture to carry out the MF method to enhance prediction accuracy, called deep matrix factorization. Experimental results of the proposed approach are positive when we perform on the published educational dataset.

Keywords: Educational data mining · Deep learning · Matrix factorization · Courses recommendation · Student performance prediction

T. K. Dang et al. (Eds.): FDSE 2021, LNCS 13076, pp. 408–423, 2021.
https://doi.org/10.1007/978-3-030-91387-8_26

1 Introduction

The original meaning of designing and developing such systems was the vision that Artificial Intelligence (AI) could give a promising solution to the limitations educational professionals face. Challenges in education include optimizing the faculty to student ratio, classifying students to improve individual student performance, and predicting student learning accurately to give appropriate suggestions for personalized students. It is possible to apply the advantages of current computer science development as classification methods, machine learning, decision-making recommendation system. Thus, the study [1] gave a systematic review and assessed the impact of AI on education. It is a qualitative research study, leveraging literature review as a research design and method was used.

AI researchers were keenly seeking a meaningful venue for their enthusiasm to spread the power of AI in many traditional fields when AI was blossoming. Computer scientists, cognitive scientists, educational professionals viewed the newborn Intelligent Tutoring Systems (ITS) to fulfil their various goals. In the paper [2], the authors reviewed the historical survey of ITS development. ITS uses AI techniques and support quality learning for individuals with little or no human assistance. As a result, ITS research is a multidisciplinary effort. It requires seamless collaborations of various disciplines, such as education, cognitive science, learning science, and computer science. In which (i) Artificial Intelligence (Computer Science) addresses how to reason about intelligence and thus learning, (ii) Psychology (Cognitive Science) tackles how people think and learn, and (iii) Education focuses on supporting teaching/learning.

Although the ITSs have diverse structures, the principal structure of an ITS contains four components such as Student-Model, Tutoring-Model, Domain-Model, and User-Interface. Although student modelling exists as one of four major components in the classic architecture of ITSs, it is a vital component in any ITSs [3]. It observes student behaviours in the tutor and creates a quantitative representation of student properties of interest necessary to customize instruction, respond effectively, engage students' interest, and promote learning. To ensure the student model's positive feedback feature, predicting student performance (PSP) is first researched.

Many types of studies may be using the learner's behaviour or learner grade [4]. Using learner behaviour is an implicit method in which researchers can predict student performance by observing the student's learning activities through the application system. Nevertheless, using the grade or mark of students is an explicit and straightforward method because all schools have a student grading system. Therefore, this method is widely used and in this article too.

The principal concern was that the effectiveness of advising improves with small students to teacher ratios. New expected performance prediction techniques are necessary to learning planning and predicting the risk of failing or dropping a class to solve the student retention problem. In [5], Personalized multi regression and matrix factorization approaches based on recommender systems, initially developed for e-commerce applications, accurately forecast students' grades in future courses as well as on in-class assessments. Accordingly, [6] gave the personalized prediction much more effective than the general rule prediction for the whole group of students. Their study was conducted with 772 students registered in e-commerce and e-commerce technologies modules at

higher educational institutions. The study aimed to predict student's overall performance at the end of the semester using video learning analytics and data mining techniques. So that, the recommender system for personalized advising is better than the traditional data mining for suggesting general prediction. Many researchers address the student performance prediction using the recommender system, but these approaches still need improvement.

In the applications of artificial intelligence in general and machine learning in particular, it is essential to have a large enough dataset to mine well. Fortunately, there is the educational data mining competition [7], and registration for the competition and the dataset was entirely free, in line with the goals of promoting educational data mining. Even though the competition has already been finished, interested researchers can still get the dataset with permission. So, many studies in educational data mining used this dataset (including this study) because of its usefulness.

Recently, Deep Learning (DL) has outperformed well-known Machine Learning (ML) techniques in many domains, e.g., cybersecurity, natural language processing, bioinformatics, robotics and control, and medical information processing, among many others. [8] propose a more holistic approach to provide a more appropriate starting point to develop a complete understanding of DL. They outline the importance of DL and present the architecture of DL techniques and networks, and we can apply the deep learning architecture to other methods.

It is possible to apply these architectures to other methods. [9] has applied deep learning architecture into matrix factorization for improving the prediction accuracy in the entertainment field. Each method will be suitable for a specific problem and specific data samples. The research had a positive result in entertainment so that it can be good perform in education.

2 Related Works

Many research addresses predicting student performance by several data mining methods. However, each applied method has both advantages and disadvantages. Therefore, in [10], The survey synthesizes the intelligent models and paradigms applied in education to predict student performance. The survey identifies several key challenges and provides recommendations for future research in educational data mining. They proposed many traditional data mining algorithms to start-of-the-art methods such as statistical models, neural networks, Tree-based models, Bayesian-based models, Support Vector Machines, Instance-based models, and others.

The author of the paper [11] listed and compared implementing three different decision tree algorithms. They showed that J48 is the best decision tree algorithm used as a prediction and classification road map of students' actions. Additionally, decision tree graphs were affected by the number of input attributes and the end class attributes. Another work used k-NN and decision tree classification methods to predict employee performance on the internal dataset. Many comparative results are also interested in and studied by researchers. For example, [12] compared two methods, Decision Tree and Bayesian Network algorithms, to predict student academic performance.

The intelligent course recommendation system uses association rules to recommend courses to the student by common rules; however, this system is not personalized for

each student. Moreover, Huu-Quang Nguyen et al. [13] have used the sequential rules algorithm applied to predicting student performance to give suggestions for students to choose elective courses. Another study proposed a system for academic advising using case-based reasoning (CBR) that recommends the student the most suitable major in his case after comparing the historical case with the student case [14].

In [15], the authors focus on designing a recommender system that recommends a set of learning objects to multiple students. Moreover, to deal with multi-decision group recommendations, they model the recommendation process as a non-cooperative game to achieve Nash equilibrium and demonstrate the effectiveness of their proposed model with a case study experiment. Furthermore, they built the system to help university students choose elective courses using a hybrid multi-criteria recommendation system with genetic optimization [16].

Rivera A.C. et al. [17] had a systematic mapping study about education recommender systems (RS). Thus, they have statistics several methods to address the problem of predicting student performance by using RS. In the paper [18], the author's proposed methods can build course recommendation systems, such as user/student k-nearest neighbours (student-kNN), item/course-kNN, standard MF, and biased MF. These methods are analyzed and validated using an actual data set before selecting the appropriate methods. They presented the framework for building the course recommendation system. However, this study focuses on the application systems and uses baseline methods.

Several works considered integrating social networks into RS, e.g., [19] have shown that the prediction accuracy can be improved by utilizing users' social networks in many ways. They have compared methods to integrate social networks into the MF. Recently, there has been a rapidly growing amount of online social networks like Facebook, Twitter. Many researchers have increasingly considered approaches for a recommendation based on social networks because they believe they affect each other. Several experiments confirmed that the social network provides independent sources of information, which can be exploited to improve the quality of recommendations. Indeed, [20] discovered that the relationship between classroom members is integrated into the training model, making the prediction better accurate. However, the algorithm is restricted only to used for data sets with user relationships. Additionally, the paper [21] proposed an approach to gather the relationships of the courses (e.g., knowledge/skills) and use those relationships to integrate into the Matrix Factorization to solve the PSP problem in the ITS.

Likewise, in the paper [22], the authors proposed to exploit multiple relationships by using multi-relational factorization models (MRMF) to improve accuracy for the PSP problems in Student-Model. However, these methods have not taken advantage of social relationships that can be integrated. In [23], the authors proposed an approach that aims to provide a solution to student performance predicting problems in ITS by combining Multiple Linear Regression (modelling Emotional Impact) and a Weighted MultiRelational Matric Factorization model advantage both students cognitive and emotional faculties. Their method considers the relationships that exist between students, tasks and skills, and their emotions.

In recent years, it has been common to transfer knowledge from one domain to another has gained much consideration among scientists. Tsiakmaki M. et al. [24] used transfer learning. A machine learning approach (deep neural networks) aims to exploit

the knowledge retrieved from one problem to improve the predictive performance of a learning model. Likewise, in the study [25], the authors proposed deep learning models (Long Short Term Memory and Convolutional Neural Networks) to predict the student performance prediction problem in educational data mining. They used some techniques for data pre-processing (e.g., Quantile Transforms, MinMax Scaler) before fetching them into deep learning models and robust machine learning such as Linear Regression to do prediction tasks.

This study will be introducing applying deep learning architecture into matrix factorization technology, improving predicting student performance. First, we present an overview of problem definitions that can predict student's marks. Then, we introduce baseline methods such as matrix factorization and biased-matrix factorization. Next, the approach of integrating the deep learning architecture is conducted. Last, the result and the comparison of this study were presented.

3 Proposed Method

3.1 Problem Definition

In the studies [18, 20, 21], the author presented the mapping the predicting student performance problem to recommendation prediction task. In recommender systems, there are three main terms, which are user, item, and rating. The recommendation task predicts the user's rating for all un-rated items and recommends top-N highest predicted scores. Similarly, the PSP problem contains three essential objects: student, course, and performance (correct/incorrect). The task predicts the course's results that the students have not learned or solved in this setting. As presented in Fig. 1, there is a similar mapping between the PSP and RS. Where student, course, and grading would become user, item, and rating, respectively.

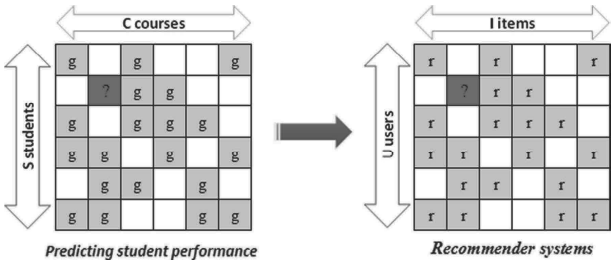

Fig. 1. The similar mapping between PSP and RS

Student scoring management systems seem to be available to all universities, but they have not exploited them effectively. The problem is that we have to use them to predict student performance by using computer science methods. Although different datasets may have diverse structures, the principal structure contains three main fields (student-id, course-id, performance).

This study used the ASSISTments dataset, a web-based math tutoring system, first created in 2004 as joint research conducted by Worcester Polytechnic Institute and Carnegie Mellon University. Its name, ASSISTments, came from the idea of combining assisting the student with the automated assessment of the student's proficiency at a fine-grained level [7]. Thousands of middle and high school students use ASSISTments for their daily learning, homework, and preparing the MCAS (Massachusetts Comprehensive Assessment System) tests. 2010–2011, more than 20,000 students and 500 teachers used the system, which was considered part of their regular math classes in and out of Massachusetts. The snapshot of the ASSISTments dataset is displayed in Fig. 2. Some fields are necessary for mining, such as "User_id", "Problem_id", and "Correct".

	A	B	C	D	E	F	G
1	user_id	student_class_id	assignment_id	assistment_id	problem_id	skill_id	correct
2	77759	12138	245748	12914	12914	231	1
3	77759	12138	245748	15320	15320	231	1
4	77759	12138	245748	14529	14529	231	1
5	77912	12138	245698	1159	1159	100	0
6	77912	12138	245698	1647	1647	93	1
7	77912	12138	245698	2705	2705	100	1
8	77912	12138	245698	2186	2186	35	1
9	77912	12138	245698	1653	1653	35	1

Fig. 2. A snapshot of the data sample

Figure 3 shows an example of how we can factorize the students and problems (the performance is correct/1 or incorrect/0). From this point to the rest of the paper, we call course, problem, task interchangeably.

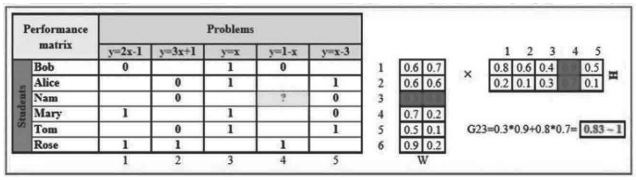

Fig. 3. An example of factorizing on students and problems

For improving the accuracy of prediction, many researchers integrated some information from independent sources. These studies have a better result.

3.2 Baseline Methods

Recommender systems are typically used by a list of recommendations using collaborative filtering (CF), content-based filtering, or a hybrid approach. Collaborative filtering methods are classified as memory-based and model-based collaborative filtering. A well-known example of memory-based processes is user-based algorithms and item-based

algorithms. Model-based approaches are the latent factor models, especially the matrix factorization method.

Matrix Factorization Method

Matrix Factorization is an effective method for latent factor models. The matrix factorization is a flexible model in dealing with various datasets, applications, and fields. Approximating a matrix $X \in R^{|S|\times|C|}$ by a product of two smaller matrices, W and H, is the main idea of matrix factorization. Figure 4 below describes a model that factorized the matrix and gives a predicted grading for the student to learn a course.

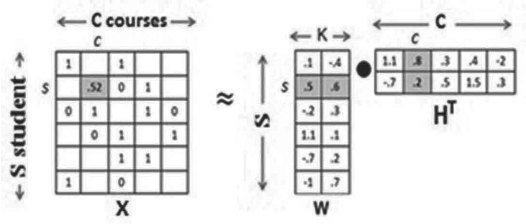

Fig. 4. The prediction process of matrix factorization

Based on [18], This method can be formalized as follows. The predicting student performance is dealing with s students and c courses, whose grades are collected in a matrix $X = \left(g_{s,c}\right) \in G_{s\times c}$ where $g_{s,c}$ is the grading that the student s learnt the course c (typically, float values between 0 and 1), or $g_{s,c} = \varnothing$ if the student s has not learnt the course c. We look for a factorization of G of the form $G \approx W \cdot H^T$. Where W is a $s \times k$ matrix, and H is a $k \times c$ matrix. Here, W, H are seen as the projection/co-projection of the s students and c courses into a k-dimensional latent space. Let w_{sk} and h_{ck} are the elements of two matrices W and H, respectively. To predict the grade/mark g for a student s to study a course c:

$$\widehat{g_{sc}} = \sum_{k=1}^{K} w_{sk} h_{ck} = w_s h_c^T \tag{1}$$

Root Mean Square Error (RMSE) is a criterion to find optimal values for the parameters W and H. It is determined:

$$RMSE = \sqrt{\frac{1}{|\mathrm{D}^{test}|_{s,c,g\in\mathrm{D}^{test}}} \sum \left(g_{si} - \widehat{g_{si}}\right)^2} \tag{2}$$

In the MF technique [18], training the model is to find the optimal parameters W and H. These matrices are initialized with some random values (from the normal distribution). Besides, the error function is added as a term for preventing over-fitting. The error function is determined:

$$o^{MF} = \sum_{(s,c,g)\in\mathrm{D}^{train}} \left(g_{sc} - \sum_{k=1}^{K} w_{sk} h_{ck}\right)^2 + \lambda\left(\|W\|_F^2 + \|H\|_F^2\right) \tag{3}$$

Where $\|\cdot\|_F^2$ is a Frobenius[1] norm, λ is a regularization weight. The error function O^{MF} can be derived to w_s and h_c resulting in the following updated rules for learning the model parameters. The w_{sk} and h_{ck} are updated by the equations below (where$e_{sc} = g_{sc} - \widehat{g}_{sc}$, and $w_{sk}^{'}$ is the updated value ofw_{sk}, and $h_{ck}^{'}$ is the updated value of h_{ck}) The values of $w_{sk}^{'}$ and $h_{ck}^{'}$ are carried out respectively.

$$w_{sk}^{'} = w_{sk} + \beta(2e_{sc}h_{ck} - \lambda w_{sk}) \tag{4}$$

$$h_{ck}^{'} = h_{ck} + \beta(2e_{sc}w_{sk} - \lambda h_{ck}) \tag{5}$$

Where β is the learning rate. We update the values of W and H iteratively until the error converges to its minimum $\left(O_{n-1}^{MF} - O_n^{MF} < \varepsilon\right)$ or reaching a predefined number of iterations. Finally, the performance of student s on courses c is now determined by Eq. (6) and Fig. 4*:*

$$\widehat{g_{sc}} = \sum\nolimits_{k=1}^{K} w_{sk}h_{ck} = w_s h_c^T \tag{6}$$

Biased Matrix Factorization Method

We have presented the standard matrix factorization to encode the student/course latent factors. Now, we introduce how to use the biased matrix factorization (BMF) to deal with the problem of "user effect" ("user bias") and "item effect" ("item bias") [18]. The user and item biases are on the educational setting, respectively, the student and course biases/effects. The student effect (student bias) models how good/clever/bad a student is (i.e., how likely is the student to perform a course correctly), and the course effect (course bias) models how difficult/easy the course is (i.e., how likely is the course to be performed correctly). With these biases, the prediction function for student s on course c is presented by

$$\widehat{g}_{sc} = \mu + b_s + b_c + \sum\nolimits_{k=1}^{K} w_{sk}h_{ck} \tag{7}$$

$$\boldsymbol{\mu} = \frac{\sum(\boldsymbol{s}, \boldsymbol{c}, \boldsymbol{g}) \in \boldsymbol{D^{train}}}{\left|\boldsymbol{D^{train}}\right|} \tag{8}$$

$$b_s = \frac{\sum_{\left(s', c, g\right) \in \boldsymbol{D^{train}} \left| s' = s \right.} (\boldsymbol{g} - \boldsymbol{\mu})}{\left|\left\{\left(\boldsymbol{s}', \boldsymbol{c}, \boldsymbol{g}\right) \in \boldsymbol{D^{train}} | \boldsymbol{s}' = \boldsymbol{s}\right\}\right|} \tag{9}$$

$$b_c = \frac{\sum_{\left(s, c', g\right) \in \boldsymbol{D^{train}} \left| c' = c \right.} (\boldsymbol{g} - \boldsymbol{\mu})}{\left|\left\{\left(\boldsymbol{s}, \boldsymbol{c}', \boldsymbol{g}\right) \in \boldsymbol{D^{train}} | \boldsymbol{c}' = \boldsymbol{c}\right\}\right|} \tag{10}$$

Moreover, the error function is also changed by adding these two biases to the regularization:

$$o^{BMF} = \sum\nolimits_{(s,c,g) \in D^{train}} \left(g_{sc} - \mu - b_s - b_c - \sum\nolimits_{k=1}^{K} w_{sk}h_{ck}\right)^2 + \lambda\left(\|W\|_F^2 + \|H\|_F^2 + b_s^2 + b_c^2\right) \tag{11}$$

[1] https://en.wikipedia.org/wiki/Matrix_norm#Frobenius_norm.

3.3 Deep Learning Matrix Factorization Method

A current trend in the field of predicting student performance is to improve the quality of the predictions utilizing different techniques of MF, such as Biased-MF [18], Social-MF [20], and CourseRelationship-MF [21]. However, all these techniques rely on the same approach: they pose an optimization problem through an error function by integrating more information that measures the divergence of the model, and the model with lower errors is better.

The main idea of integrating deep learning architecture into the MF technique is recursing the matrix factorization, repeatedly approximate the output matrix until it meets the best result. Each step in the process is a baseline method (matrix factorization); the complete model summarises the stack of the models in stages.

We are trying to break with the MF paradigm, and based on the paper [9], we present an integrating model that uses the DL principles to refine the model's output through successive training. It is called the Deep Learning Matrix Factorization (DLMF). Figure 5 illustrates the operation of DLMF. As we can observe, the model is initialized with the standard input of a CF-based RS: a matrix X that contains the student's grading/score/mark to the course/problem/exercise.

As in the classical MF method, this matrix X will also be called $X = G^0$ which is the beginning of the process. Approximating a matrix $X \in G^{|S|\times|C|}$ by a product of two smaller matrices, W^0 and H^0, is a form $\widehat{G}^0 = W^0 \cdot H^0$. The matrix $\widehat{G}^0$ provides all the predicted gradings, and they were stored in the stack at the first step. At this step, the recursive is begins. A new matrix $G^1 = X - \widehat{G}^0$ is built by computing the attained errors between the original gradings matrix X and the predicted gradings stored in $\widehat{G}^0$. A factorization again approximates this new matrix G^1 into two new small rank matrices $\widehat{G}^1 = W^1 \cdot H^1$, which produces the errors at the second step $G^2 = G^1 - \widehat{G}^1$. This process is repeated many times by generating and factorizing successive error matrices G^1,…,G^T. Presumably, this sequence of error matrices converges to zero, so we get preciser predictions as we add new layers to the model.

Similar to the standard MF method was presented at the baseline methods section above. Two small k-rank matrices are trained such that the product $W^0 \cdot H^0$ is a good approximation of the rating matrix $G^0 = X$, that is, in the usual Euclidean distance. The term $W^0 \in G^{0|s|\times|k^0|}$ is a matrix where each row s is a vector w_s (rendering the student s) and has k^0 latent factors. Similarly, the term $H^0 \in G^{0|c|\times|k^0|}$ is a matrix where each row c is a vector h_c (rendering the course c) and has k^0 latent factors. The approximation can be expressed as follows:

$$G^0 \approx \widehat{G}^0 = W^0 \cdot H^0 \tag{12}$$

To implement the deep learning model, we subtract the approximation performed by $\widehat{G}^0$ to the original matrix X, to obtain a new sparse matrix G^1 that contains the prediction error at the first iteration:

$$G^1 = X - \widehat{G}^0 = G^0 - W^0 \cdot H^0 \tag{13}$$

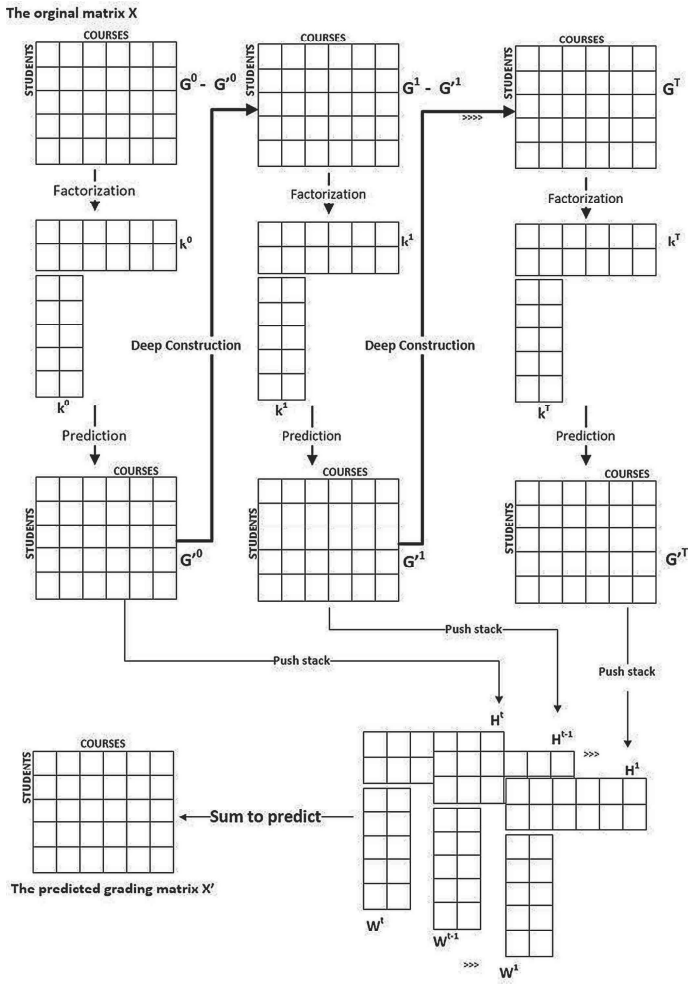

Fig. 5. Graphical model of the deep learning matrix factorization technique for PSP

Note that positive values in the matrix G^1 mean that the prediction is low and need to be increased. Similarly, the negative values in the matrix G^1 mean that the prediction is high and need to be decreased. Indeed, this adjustment is the main idea of applying the deep learning approach. To do this, we need to perform a new factorization to the error matrix G^1 in such a way that

$$G^1 \approx \widehat{G}^1 = W^1 \cdot H^1 \tag{14}$$

The approximation process in each step is performed similarly. However, two matrices W^1 and H^1 have orders $s \times k^1$ and $k^1 \times c$ for a definite number of latent factors k^1. Note that we should take $k^1 \neq k^0$ to get various resolutions in the factorization.

In the general case, if we computed at $t-1$ steps of the deep learning procedure, the t^{th} matrix of errors can be determined:

$$G^t = G^{t-1} - \widehat{G}^{t-1} = G^{t-1} - W^{t-1} \cdot H^{t-1} \tag{15}$$

At the step t, we are also factorizing into matrices W^t and H^t have the k^t latent factors until the sequence of error matrices converge to zero.

$$G^t \approx \widehat{G}^t = W^t \cdot H^t \tag{16}$$

Once the deep factorization process ends after T steps, the original grading matrix X can be reconstructed by adding the estimates of the errors as

$$\begin{aligned} &X \approx \widehat{X} = \widehat{G}^0 + G^1 = \widehat{G}^0 + \widehat{G}^1 + G^2 = \cdots = \widehat{G}^0 + \widehat{G}^1 + \widehat{G}^2 + \cdots + \widehat{G}^T = W^0 \cdot \\ &H^0 + W^1 \cdot H^1 + \cdots + W^T \cdot H^T = \sum_{t=0}^{T} W^t \cdot H^t \end{aligned} \tag{17}$$

For any step $t = 0, \cdots, T$ the factorization $G^t \approx W^t \cdot H^t$ is sought by the standard method of minimizing the euclidean distance between G^t and $W^t \cdot H^t$ by gradient descent with regularization. The error function for the DLMF now becomes:

$$o^{DLMF} = \sum_{t=0}^{T} \left(\sum_{(s,c,g) \in \mathrm{D}^{t\,train}} \left(g_{sc}^t - \sum_{k^t=1}^{K} w_{sk^t}^t h_{k^t c}^t \right)^2 + \lambda^t \left(\|W^t\|_F^2 + \|H^t\|_F^2 \right) \right) \tag{18}$$

Where the term λ^t is the regularization hyper-parameter of the step t to avoid overfitting.

The error function O^{DLMF} can be derived to w_s^t and h_c^t resulting in the following updated rules for training the model parameters. The w_{sk}^t and h_{ck}^t are updated by the equations below (where $e_{sc}^t = g_{sc}^t - \hat{g}_{sc}^t$, and ${w_{sk}^t}'$ is the updated value of w_{sk}^t, and ${h_{ck}^t}'$ is the updated value of h_{ck}^t). With the new error function of the DLMF, The values of ${w_{sk}^t}'$ and ${h_{ck}^t}'$ are updated respectively

$${h_{ck}^t}' = h_{ck}^t + \beta^t \left(2 e_{sc}^t w_{sk}^t - \lambda h_{ck}^t \right) \tag{19}$$

$${w_{sk}^t}' = w_{sk}^t + \beta^t \left(2 e_{sc} h_{ck}^t - \lambda w_{sk}^t \right) \tag{20}$$

Where β^t is the learning rate hyper-parameter of step t to control the learning speed.

In this way, after finishing the nested factorization, all the predicted ratings are collected in the matrix $\widehat{X} = \left(\hat{g}_{sc} \right)$, where the predicted the grading of the student s to the course c is given by

$$\hat{g}_{sc} = \sum_{t=0}^{T} \sum_{k=1}^{K} w_{sk}^t h_{kc}^t = \sum_{t}^{T} w_s^t * \left(h_c^t \right)^T \tag{21}$$

Note that this method consists of successive repetitions of an MF process using the results of the previous MF as input. All the hyper-parameters are stored in the stack, so we may easily use a recursive approach and recursive implementation algorithm.

Proposed Algorithm

The algorithm receives inputs as the original matrix X and the model hyper-parameters. Similarly, the output of the algorithm will be a stack containing the pairs $\langle W, H \rangle$ that fully represent the predictability of the deep learning process.

Note that these hyper-parameters were stacked so that each of the factorizations performed uses different hyper-parameters. The hyper-parameters of the first factorization will be pushed at the top of the stack, and the hyper-parameters of the second factorization in the next one. And so on until the parameters of the last factorization will be pushed at the bottom of the stack. This allows us to define the stopping criteria of the algorithm as the depth of stack, which is usually around four layers.

Details of the proposed method that integrates the deep learning architecture into Matrix Factorization are presented in the function below "Deep-Learning-Matrix-Factorization – DLMF". This DLMF is recursively factorizing student and course using the stacks of stochastic gradient descent with k latent factors, β learning rate, λ regularization weight, stopping condition, and the depth. For example, we pop the hyper-parameters to carry out the block of MF statements in each depth. In each MF statement block in lines 1–13, we perform as standard MF method. Then we recursive call and push the complete training models to the stack in lines 18–19.

```
Function Deep-Learning-Matrix-Factorization
Input:D^train, K, betas, lamdas, stopping condition, depth
Output: W, H
1. Let s ∈ S be a student, c ∈ C a course, g ∈ G a grade
2.   Let  W[|S|][K], H[|C|][K]  be latent factors of
students, courses
3.   W ← N(0,σ^2) and H ← N(0,σ^2)
4.   k, t, β, λ ← pop (K, T, Betas, Lamdas)
5.   while (the stopping condition is NOT met) do
6.        for each (s,c,g_sc) from D^train
7.          ĝ_sc ← Σ_k^K (W[s][k] * H[c][k])
8.          e_sc = g_sc − ĝ_sc
9.          for  k = 1..K do
10.           W[s][k] = W[s][k] + β * (2e_si * H[c][k] − λ * W[s][k])
11.           H[c][k] ← H[c][k] + β * (2e_sc * W[s][k] − λ * H[c][k])
12.         end for
13.       end for
13.   end while
14.   if (is-empty(K)) return new stack (⟨W,H⟩)
15.   else
16.       Ĝ = G − W · H
17.       Params-Stack = Deep-Learning-Matrix-
Factoriztion(Ĝ, K, T, Betas, Lamdas)
18.       Return push(⟨W,H⟩, Params-Stack)
19.   end else.
20.   end function.
```

4 Result

4.1 Dataset

The ASSISTments[2] dataset is published by the ASSISTments Platform. It is a web-based tutoring system that assists students in learning mathematics and gives teachers an assessment of their students' progress. It allows teachers to write individually, and each ASSISTments is composed of questions and associated hints, solutions, web-based videos. After preprocessing, this dataset contains 8519 students (users), 35978 tasks (items), and 1011079 gradings (ratings).

4.2 Evaluation

In this work, predicting student marks is the task of rating prediction (explicit feedback), so we use a popular measure in RS is Root Mean Squared Error (RMSE), for model evaluation. We have used the hold-out approach (2/3 of data is used for training, and 1/3 of data is used for testing) for experimenting with the models.

The accuracy of the prediction depends on the parameters that feed to the algorithm. If the parameters were unsuitable, the prediction accuracy would not be good even though the algorithm is correct. Thus, finding the best parameter is significant.

The hyper-parameters search, a searching parameter method, is applied to search all the parameters of the approached models [18]. The hyper-parameters search has two stages based on grid search: raw search (for the long segments) and smooth search (for the short segments). First, the raw search stage is carried out to find the best hyper-parameters in the long data segments. Then, we perform a smooth search to find the nearby best hyper-parameters. For example, using RMSE as a criterion, the hyper-parameter search results for the models on the ASSISTments dataset are presented in Table 1.

Table 1. Hyper-parameters on ASSISTments dataset

Methods	Hyper parameter
MF	$\beta = 0.03$, #iter $= 50$, K $= 4$, $\lambda = 0.05$
BMF	$\beta = 0.0015$, #iter $= 50$, K $= 2$, $\lambda = 0.1$
DLMF	Deep (T) $= 4$ numFactors (K) $= [3, 6, 3, 3]$; numIters (Iter) $= [50, 50, 50, 50]$; learningRate (β) $= [0.01, 0.1, 0.1, 0.1]$; regularization (λ) $= [0.01, 0.1, 0.1, 0.01]$)

After having the best hyper-parameters, we use them for training and testing each respective model. However, the training time is slower than without deep learning. How much deeper training, the time delay that many times.

[2] https://sites.google.com/site/assistmentsdata/home.

4.3 Experimental Result

We have compared integrating deep learning architecture into matrix factorization (DLMF) to predict student performance in the ITS with other methods, such as standard MF, BMF, and DLMF. Fortunately, many open-source libraries implemented these algorithms, such as LibRec (librec.net), MyMediaLite (mymedialite.net), Collaborative Filtering For Java (CF4J), that we can inherit from them.

We conducted three experiments; the experimental results are displayed in Fig. 6. Comparing with others, the RMSE of the proposed approach (DLMF) is the smallest one (0.419) on the dataset. The smallest error demonstrates the best model.

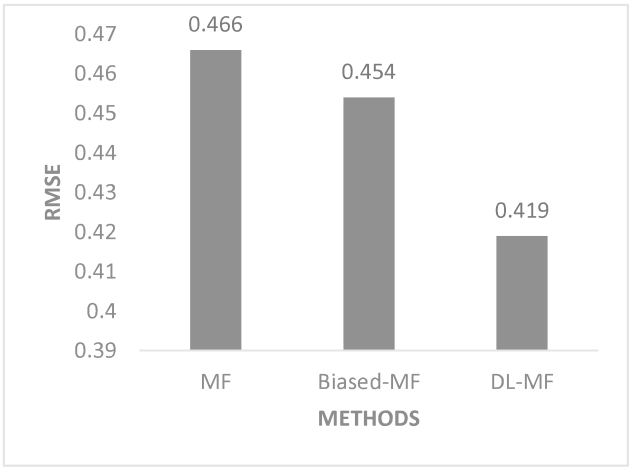

Fig. 6. Experiment results on the ASSISTment dataset

5 Conclusion

This paper has introduced an approach that utilizes the deep learning architecture to carry out the MF method to enhance the accuracy of student performance prediction. We can take advantage of the deep learning principles to refine the model's output (predicted matrix) through successive training for building the prediction model with this approach. Thus, the prediction results can be improved significantly. Conducting experiments on the published competition datasets shows that the proposed process works well.

Applying deep learning architecture to matrix factorization make the training time will be slow. However, it is easy for us to implement a parallel algorithm to solve this problem. This work uses deep learning architecture for the standard matrix factorization without using other complex integrating techniques. Future research is how to find the meta-data for integrating to get highly effective and fast algorithms.

References

1. Chen, L., Chen, P., Lin, Z.: Artificial intelligence in education: a review. IEEE Access **8**, 75264–75278 (2020). https://doi.org/10.1109/ACCESS.2020.2988510
2. Guo, L., Wang, D., Gu, F., Li, Y., Wang, Y., Zhou, R.: Evolution and trends in intelligent tutoring systems research: a multidisciplinary and scientometric view. Asia Pac. Educ. Rev. **22**(3), 441–461 (2021). https://doi.org/10.1007/s12564-021-09697-7
3. Khodeir, N.: Student modeling using educational data mining techniques. In: ACCS/PEIT 2019 - 2019 6th International Conference on Advanced Control Circuits and Systems (ACCS) & 2019 5th International Conference on New Paradigms in Electronics & Information Technology, pp. 7–14 (2019). https://doi.org/10.1109/ACCS-PEIT48329.2019.9062874
4. Bogarín, A., Cerezo, R., Romero, C.: A survey on educational process mining. Wiley Interdiscip. Rev. Data Min. Knowl. Discov. **8**, e1230 (2018). https://doi.org/10.1002/widm.1230
5. Elbadrawy, A., Polyzou, A., Ren, Z., Sweeney, M., Karypis, G., Rangwala, H.: Predicting student performance using personalized analytics. Computer (Long. Beach. Calif). **49**, 61–69 (2016). https://doi.org/10.1109/MC.2016.119
6. Hasan, R., Palaniappan, S., Mahmood, S., Abbas, A., Sarker, K., Sattar, M.: Predicting student performance in higher educational institutions using video learning analytics and data mining techniques. Appl. Sci. **10**(11), 3894 (2020). https://doi.org/10.3390/app10113894
7. Patikorn, T., Baker, R.S., Heffernan, N.T.: ASSISTments longitudinal data mining competition special issue: a preface. J. Educ. Data Min. **12**, i–xi (2020). https://doi.org/10.5281/ZENODO.4008048
8. Alzubaidi, L., et al.: Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J. Big Data **8**(1), 1–74 (2021). https://doi.org/10.1186/s40537-021-00444-8
9. Lara-Cabrera, R., González-Prieto, Ángel., Ortega, F.: Deep matrix factorization approach for collaborative filtering recommender systems. Appl. Sci. **10**(14), 4926 (2020). https://doi.org/10.3390/app10144926
10. Namoun, A., Alshanqiti, A.: Predicting student performance using data mining and learning analytics techniques: a systematic literature review. Appl. Sci. **11**(1), 237 (2020). https://doi.org/10.3390/app11010237
11. Ayyappan, G.: Ensemble classifications for student academics performance data set. Indian J. Comput. Sci. Eng. **10**, 31–34 (2019). https://doi.org/10.21817/INDJCSE/2019/V10I1/191001009
12. Ghorbani, R., Ghousi, R.: Comparing different resampling methods in predicting students' performance using machine learning techniques. IEEE Access **8**, 67899–67911 (2020). https://doi.org/10.1109/ACCESS.2020.2986809
13. Nguyen, H.Q., Pham, T.T., Vo, V., Vo, B., Quan, T.T.: The predictive modeling for learning student results based on sequential rules. Int. J. Innov. Comput. Inf. Control. **14**, 2129–2140 (2018). https://doi.org/10.24507/IJICIC.14.06.2129
14. Hasnawi, M., Kurniati, N., Mansyur, S.H., Irawati, Hasanuddin, T.: Combination of case based reasoning with nearest neighbor and decision tree for early warning system of student achievement. In: Proceedings of 2nd East Indonesia Conference on Computer and Information Technology Internet Things Ind. EIConCIT 2018, pp. 78–81 (2018). https://doi.org/10.1109/EICONCIT.2018.8878512
15. Zia, A., Usman, M.: Elective learning objects group recommendation using non-cooperative game theory. In: Proceedings of 2018 International Conference on Frontiers of Information Technology, FIT 2018, pp. 194–199 (2019). https://doi.org/10.1109/FIT.2018.00041

16. Esteban, A., Zafra, A., Romero, C.: Helping university students to choose elective courses by using a hybrid multi-criteria recommendation system with genetic optimization. Knowl.-Based Syst. **194**, 105385 (2020). https://doi.org/10.1016/J.KNOSYS.2019.105385
17. Rivera, A.C., Tapia-Leon, M., Lujan-Mora, S.: Recommendation systems in education: a systematic mapping study. In: Rocha, Á., Guarda, T. (eds.) ICITS 2018. AISC, vol. 721, pp. 937–947. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-73450-7_89
18. Thanh-Nhan, H.L., Nguyen, H.H., Thai-Nghe, N.: Methods for building course recommendation systems. Proc. - 2016 8th International Conference on Frontiers of Information Technology, KSE 2016, pp. 163–168 (2016). https://doi.org/10.1109/KSE.2016.7758047
19. Chen, R., et al.: A novel social recommendation method fusing user's social status and homophily based on matrix factorization techniques. IEEE Access **7**, 18783–18798 (2019). https://doi.org/10.1109/ACCESS.2019.2893024
20. Thanh-Nhan, H.L., Huy-Thap, L., Thai-Nghe, N.: Toward integrating social networks into intelligent tutoring systems. In: Proceedings of 2017 9th International Conference on Knowledge and Systems Engineering, KSE 2017, 2017-January, pp. 112–117 (2017). https://doi.org/10.1109/KSE.2017.8119444
21. Huynh-Ly, T.N., Le, H.T., Nguyen, T.N.: Integrating courses' relationship into predicting student performance. Int. J. Adv. Trends Comput. Sci. Eng. **9**, 6375–6383 (2020). https://doi.org/10.30534/IJATCSE/2020/320942020
22. Thai-Nghe, N., Schmidt-Thieme, L.: Multi-relational factorization models for student modeling in intelligent tutoring systems. In: Proceedings - 2015 IEEE International Conference on Knowledge and Systems Engineering, KSE 2015, pp. 61–66. Institute of Electrical and Electronics Engineers Inc. (2015). https://doi.org/10.1109/KSE.2015.9
23. Assielou, K., Théodore, C., Tra, B., Lambert, T., Daniel, K.: Emotional impact for predicting student performance in intelligent tutoring systems (ITS). Int. J. Adv. Comput. Sci. Appl. **11**(7), 219–225 (2020). https://doi.org/10.14569/IJACSA.2020.0110728
24. Tsiakmaki, M., Kostopoulos, G., Kotsiantis, S., Ragos, O.: Transfer learning from deep neural networks for predicting student performance. Appl. Sci. **10**(6), 2145 (2020). https://doi.org/10.3390/app10062145
25. 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