Grade Prediction with Neural Collaborative Filtering

1st Zhiyun Ren, 4th Huzefa Rangwala
Computer Science
George Mason University
Fairfax, VA, USA
{*zren4, rangwala*}*@gmu.edu*

2nd Xia Ning
Biomedical Informatics
The Ohio State University
Columbus, OH, USA
Xia.Ning@osumc.edu

3rd Andrew S. Lan
College of Information and Computer Sciences
University of Massachusetts Amherst
Amherst, MA, USA
andrewlan@cs.umass.edu

***Abstract*—Over the past decade low graduation and retention rates has plagued higher education institutions. To assist students in choosing a sequence of courses, choosing majors and successful academic pathways; many institutions provide several on-site academic advising services supported by data driven educational technologies. Accurate performance prediction can serve as the backbone for degree planning software, personalized advising systems and early warning systems that can identify students at-risk of dropping from their field of study.**

In this work, we present a deep learning based recommender system approach called Neural Collaborative Filtering (NCF) for predicting the grade a student will earn in a course that he/she plans to take in the next-term. Prior grade prediction methods are based on matrix factorization (MF) where students and courses are represented in a latent "knowledge" space. The deep learning inspired approach provides added flexibility in learning the latent spaces in comparison to MF approaches. The proposed approach also incorporates instructor information besides student and course information. Moreover, for proper analysis of the learned model parameters, we assume the embeddings obtained for students, courses and instructors should be non-negative. This non-negative NCF model referred by NCF$_{nn}$ model adds a rectified linear units (ReLU) on the embedding layer of NCF. The experimental results on datasets from George Mason University, a large, public university in the United States, demonstrate that the proposed NCF approaches significantly outperform competitive baselines across different test sets.

***Keywords*-Grade Prediction, Neural Network, Neural Collaborative Filtering**

I. Introduction

One of the grand challenges facing higher education is low student graduation and retention rates [19]. There is a need to develop and deploy data-driven applications and services that provide personalized advise to students pertaining to selection of degree pathways. This will allow the students to fulfill requirements pertaining to their program of study in a timely and complete manner. Many higher education institutions have implemented programs supported by educational technologies to increase such graduation rates [26]. For example, Graduation Progression Success (GPS) Advising [1] helps identify at-risk students at an early stage and was shown to successfully increase the six-year graduation rate by 6% over 4 years at Georgia State University. As another example, Academic Advising service [2] provides effective student-centered advising at Purdue University. An effective way to assist academic advising is to capture and leverage the influences from different factors on student's academic performance. In this work, we focus on predicting the grade a student may earn in a course that he/she plans to enroll for in the future i.e., next-term.

Over the past few years, several approaches inspired from the recommender systems literature have been adapted for predicting next-term student performance [5], [27]. In particular, matrix factorization (MF) approaches inspired from Collaborative Filtering decompose the given student-course grade matrix into two low-rank matrices that represent the latent "knowledge space" of courses and students. The prediction of a student's grade on an untaken course is then given by a dot product of the corresponding vectors in the two decomposed matrices [12], [20]. MF approaches have been suitable for dealing with sparse datasets [17] and their extensions have incorporated temporal and dynamic information [13] that have shown an improvement in terms of model performance. However, when calculating inner product with MF, there are certain limitations [10]. For example, such inner products (linear combination of multiplication of latent factors) may not capture complex/nonlinear relations among data. To tackle this problem, He *et al.* [10] proposed a Neural Collaborative Filtering (NCF) model, which generalizes matrix factorization and learns non-linear relationships and aims to predict the rating a user would give to an item. Specifically, NCF uses two one-hot encoded vectors of users and items as input and learns embedding features for these two elements, followed by a fully connected feed-forward neural network to predict the user's rating on the item. Empirical results on several recommender system benchmarks demonstrates that the NCF approach greatly outperforms other methods [10].

We also extend NCF with non-negativity constraints and develop a non-negative neural collaborative filtering method, denoted as NCF$_{nn}$. We apply NCF and NCF$_{nn}$ on next-term grade prediction problem. Similarly as in NCF model, we

[1] http://giving.gsu.edu/student-success/

[2] http://www.purdue.edu/advisors/index.html

978-1-7281-4493-1/19/$31.00 ©2019 IEEE
DOI 10.1109/DSAA.2019.00014

consider three elements as input: (i) students, (ii) courses and (iii) instructors. Adding of course instructor information has shown to improve the performance of prior grade prediction models [22]. Non-negative constraints on learned parameters provides for interpretability [6]. We propose NCF_{nn} by adding Rectified Linear Units (ReLU) [8] on the embedding layer. A ReLU is a unit that exploits the rectifier [18], which is an activation function that preserves the non-negative part of the argument and returns zero for the negative part.

Our experimental results on a dataset from George Mason University demonstrates that the proposed methods significantly outperform other competitive baselines for the task of grade prediction. In addition, we analyzed the model performance with respect to different embedding dimensions for students, courses and course instructors, respectively. The main contributions of our work in this paper can be summarized as follows:

1) We apply Neural Collaborative Filtering (NCF) on next-term grade prediction which, to our knowledge, has not been done before.
2) We propose NCF_{nn} model and our experimental results show that NCF_{nn} is able to gain better results than NCF model.
3) We have conducted extensive experiments on different testing sets. We provide comprehensive comparison on NCF models with different student, course and course instructor embedding dimensions. Moreover, we show detailed analysis on how non-negativity constraint affects model performance.

II. Related Work

A. Grade Prediction Approaches

Methods originating from Recommender System (RS) research have attracted increasing attention in educational data mining [6], [29]. Sweeney *et. al.* [27], [28] applied several recommender system approaches to predict next-term grade performance. The authors implement methods including SVD, SVD-kNN and Factorization Machine (FM), and provide comprehensive analysis on the prediction results. Elbadrawy *et. al.* [4] developed a domain-aware grade prediction method with student/course-group based biases. To predict student *s*' grade on course *c*, this method groups students based on student majors and academic levels; groups courses based on course levels and subjects. Then the proposed method employed group-based biases for both student and course, and obtained great improvement on grade prediction results.

B. Deep Learning in Educational Data Mining

Deep Learning (DL) techniques have been applied to solve many educational data mining problems. For example, Sharma *et al.* [24] proposed a composite deep neural network to predict human movement (e.g., walking, sitting) in educational videos. The proposed method first used a convolutional neural network to extract the video features, followed by a deep recurrent neural network to predict the human movement label. Klingler *et al.* [15] employed deep variational auto-encoders (VAE) on classification tasks in EDM. Specifically, the presented model makes effective use of unlabeled data to learn efficient feature embeddings for students, and significantly improved detection results of developmental dyscalculia (DD), compared to completely supervised training. Xiong *et al.* [31] introduced a recently developed model, Deep Knowledge Tracing (DKT), a pioneering algorithm that uses recurrent neural network to model student learning. The method is evaluated on various datasets compared with Factors Analysis Model and Knowledge Tracing models models, and the results show the proposed method outperforms the baselines on various datasets. In this work, we apply NCF model on grade prediction problem which, to our knowledge, has not been done before.

C. Deep Learning based Recommender Systems

In recent years, DL has become increasing popular in Recommender Systems (RS). Covington *et al.* [3] proposed a deep learning model for YouTube video recommendation. The model comprises two parts: 1) the candidate generation network and 2) the ranking network, both of which are fully connected neural network. The candidate generation network takes events from the user?s activity history on Youtube as input, and return a small set of candidate videos, and the ranking network assigns a score to each video with plenty features of both users and videos. Yang *et al.* [32] developed a deep neural architecture called Preference And Context Embedding (PACE) for point of interest (POI) recommendation. PACE jointly learns the embeddings of users and POIs to predict both user preference over POIs and various context associated with users and POIs. Wang *et al.* [30] developed a hierarchical Bayesian model called collaborative deep learning (CDL) as a novel method for RS problem. The proposed method used stacked denoising autoencoder to learn item features and jointly performed collaborative filtering for the user-item rating matrix.

Elkahky *et al.* [7] proposed a DL approach to mapping users and items to a latent space in order to tackle user modeling. The model parameters are learned by maximizing the similarity between users and their preferred items. They also introduced a multi-view DL model to jointly learn features of items from different domains and user features. The proposed model can gain good user representation with a rich feature set. He *et al.* [10] presented a general framework named Neural network-based Collaborative Filtering (NCF) in order to tackle the collaborative filtering problem in recommendation with non linearity. By replacing the inner product with a neural architecture that can learn an arbitrary function from data, NCF is able to gain significant improve-

ment over the state-of-the-art methods on recommendation results. This is the basic model we apply on grade prediction problem in this work.

III. Definitions and Preliminaries

A. Problem Definitions

All student-course grades up to term T will be represented by a matrix $G \in \mathbb{R}^{n \times m}$, where n is the number of students, m is the number of courses. Each entry in matrix G contains a tuple storing grade information for the corresponding student, course and instructor. Each tuple stores: (i) student identifier, (ii) course identifier, (iii) course instructor, and (iv) the grade. Therefore, each value in G, denoted as $g^l_{s,c}$, represents a grade that student s has achieved on a course c taught by instructor l, where $g^l_{s,c} \in (0,4]$. Specifically, we set letter grade "A+" and "A" correspond to 4.0, "A-" to 3.67, "B+" to 3.33 and so on. $g^l_{s,c} = 0$ indicates both letter grade "F" and that student s did not take the course c. To make a distinction on the two situations, we add a small value to letter grade "F". Finally, student-course grades at the t_{th} will be represented by G_t, and student-course grades up to the T_{th} term will be represented by $\mathrm{G}^T = \sum_{i=1}^{t} \mathrm{G}_i$ with size of $n \times m$.

In this paper, all vectors (e.g., $\mathbf{p}_s^\mathsf{T}$ and $\mathbf{q}_c$) are represented by bold lower-case letters and all matrices (e.g., G) are represented by upper-case letters. Row vectors are represented by having the transpose superscript$^\mathsf{T}$, otherwise by default they are column vectors. A predicted value is denoted by having a ˜ symbol. Given student-course grades up to term T, the objective is to predict grades for each student on courses that the student may consider for enrollment in the next term $T+1$.

B. Matrix Factorization based Grade Prediction

Matrix factorization methods have achieved extraordinary success in user-item rating prediction in recommender systems [23]. Such methods can as well be applied for the next-term grade prediction problem, when the student-course grade matrix is considered as the user-item rating matrix. Specifically, a grade matrix will be factorized into two low-rank matrices containing latent factors of courses and students in a common knowledge space [27]. We use $\mathbf{p}_s$ ($\mathbf{p}_s \in \mathbb{R}^k$) and $\mathbf{q}_c$ ($\mathbf{q}_c \in \mathbb{R}^k$) to represent latent factors of k dimensions for student s and course c, respectively. Thus, the grade of a student s on a course c can be predicted as

$$\tilde{g}_{s,c} = \mathbf{p}_s^\mathsf{T} \mathbf{q}_c, \tag{1}$$

Including the bias terms within the MF formulation has shown to be effective in modeling systematical biases [16]. For the grade prediction problem using MF, student and course biases can be included as follows:

$$\tilde{g}_{s,c} = \mathbf{p}_s^\mathsf{T} \mathbf{q}_c + b_s + b_c, \tag{2}$$

where b_s and b_c are bias terms for student s and course c, respectively.

C. Neural Network-based Collaborative Filtering

Fig. 1 shows the structure of NCF for rating prediction in Recommender Systems. To predict user u's rating on item i, that is, $y_{u,i}$, NCF model takes two one-hot encoded vectors for user u and item i as input. Above the input layer is the embedding layer. It is a fully connected layer that projects the sparse representation to a dense vector. The embeddings of user and item are then concatenated and sent to a multi-layer fully connected neural network, which contains neural collaborative filtering (NCF) layers. Finally, the output of NCF layers is fed into the output layer and returns the predicted rating $y_{u,i}$. At each layer, different activation functions can be added, such as sigmoid function, hyperbolic tangent (tanh), and Rectifier (ReLU). The training of the model is performed by minimizing the point-wise loss between predicted ratings and the corresponding ground-truth ratings. NCF has shown to produce better recommendation results in comparison to MF methods. Unlike the traditional MF method, the NCF structure provides several advantages in terms of flexibility of input representation. The user/item latent factors can have varying number of dimensions and additional inputs beyond user/items can be easily incorporated within the NCF model.

IV. Methods

A. NCF for Grade Prediction

In this work, we formulate the grade prediction problem within the NCF framework. Based on prior work [22] that course instructors can greatly influence student's grades, we consider three elements in grade prediction problem with NCF model, i.e., students, courses and the course instructors. Specifically, to predict student s's grade on course c taught by instructor l, we input three separate one-hot encoded vectors for the corresponding elements into our NCF model's input layer. Similar to the original NCF model, we have an embedding layer above the input layer, representing the latent factors for the three elements. Then the concatenated vector of the embeddings is fed into neural collaborative filtering layers to predict the final grade $g^l_{s,c}$. Different from original NCF, we specifically choose ReLU as activation function at each layer. As aforementioned, a ReLU exploits the activation function rectifier which preserves the non-negative part of the argument and returns zero for the negative part. Therefore, we use ReLU to achieve non-negative constraint. The structure of this model is shown in Fig. 2.

1) Rectified Linear Unit: Rectifier is an activation function that is defined as follow:

$$f(x) = max(0,x). \tag{3}$$

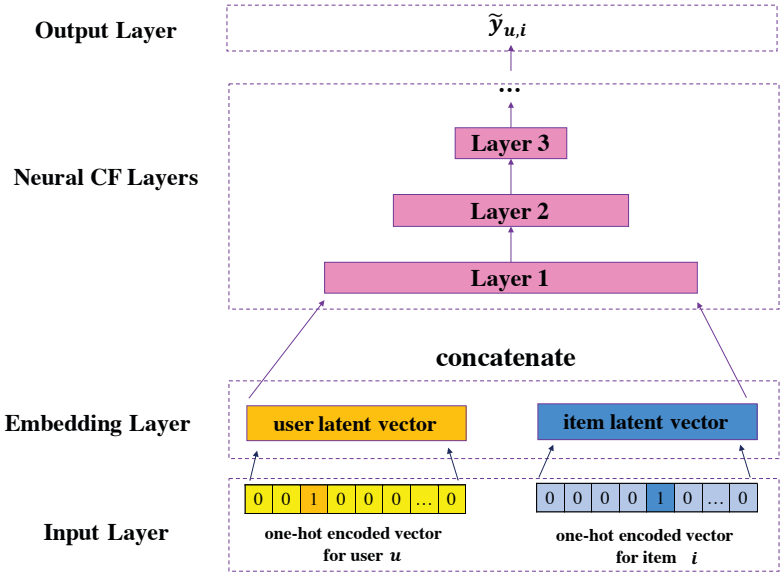

Figure 1: Model Structure of NCF on Recommender Systems

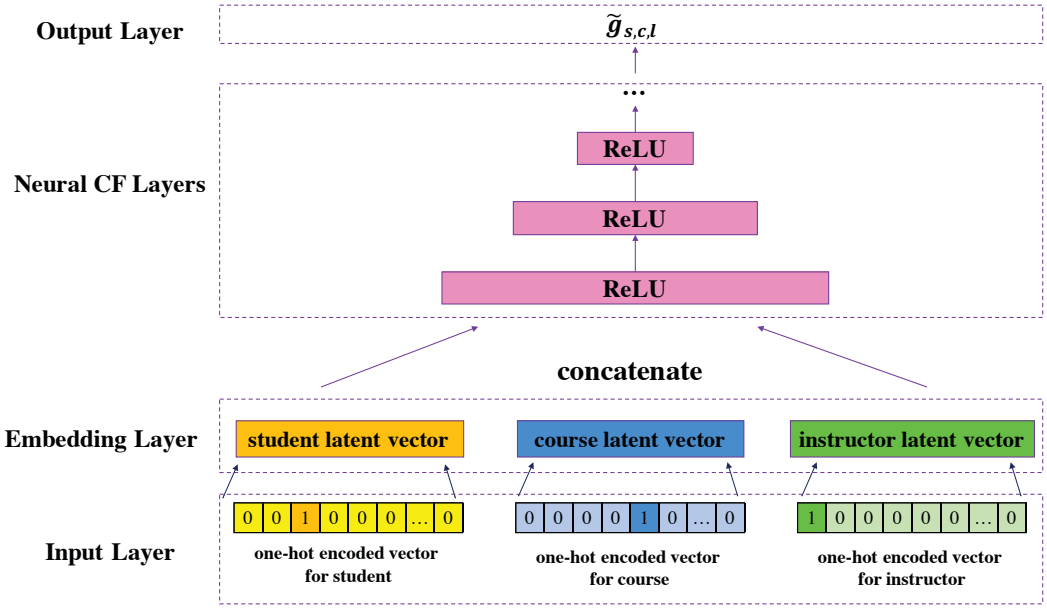

Figure 2: Model Structure of NCF on Grade Prediction

where x is the input to a neuron. A ReLU is a unit that exploits the rectifier [18].

2) NCF with Non-Negativity Constraints: Since the embeddings can be interpreted in latent knowledge spaces, in order to get proper analysis of the model, we add non-negativity constraint on the embeddings by adding ReLU on the embedding layer. Since each one of NCF layers in our model has ReLU as activation function, by adding ReLU on the embedding layer, the modified model has non-negativity values on all layers. We denote this NCF with non-negativity constraint as NCF_{nn}.

3) Parameter Learning: To predict grades in term t, the loss function for NCF and NCF_{nn} can be formulated as follows:

$$L = \sum_{g^l_{s,c} \in G^{t-1}} (g^l_{s,c} - \tilde{g}^l_{s,c})^2 \tag{4}$$

We use Adaptive Moment Estimation (Adam) [14] method to learn model parameters.

Table I: Dataset Statistics

Major	#S	#C	#S-C
MATH	209	84	2,846
PSYC	1,114	95	14,377
CHEM	342	55	4,649
CS	988	76	13,809
IT	334	82	6088
BIOL	1,629	109	21,519

#S, #C and #S-C are the number of students, courses and student-course grades from Fall 2009 to Spring 2016, respectively.

Table II: #S-C for Different Terms

Major	Fall 2009 to Fall 2014	Spring 2015	Fall 2015	Spring 2016
MATH	942	100	78	168
PSYC	6,060	595	453	749
CHEM	1,139	86	106	103
CS	3,041	413	396	683
IT	1,492	474	439	599
BIOL	5,676	577	461	749

V. Experiments

A. Dataset Description

The data used in this work is obtained from a George Mason University. The dataset was extracted in the period of Fall 2009 to Spring 2016. It includes information for 23,013 transfer students (TR) and 20,086 first-time freshmen (FTF; i.e., students who begin their study initially at this University) across 151 majors. Specifically, we evaluated the proposed models on the following six majors: (i) Mathematical Sciences (MATH), (ii) Psychology (PSYC), (iii) Chemistry (CHEM) (iv) Computer Science (CS), (v) Information Technology (IT), and (vi) Biology (BIOL). Table I presents the details of these majors. In our experiments, we only apply the models on FTF students as these students have more and complete data throughout college study than TR students. Table II shows the number of student-course grades of FTF students from different majors in different terms.

B. Experimental Protocols

We train the different models on data up to term $T-1$ and make predictions for term T. For example, we train our models using data from Fall 2009 to Fall 2015, and test our models using data from Spring 2016 (i.e., T = Sprint 2016). Fig. 3 shows our experimental protocols. We train and test our models on each major separately because different majors could have different student and course characteristics, etc. We also use different embedding dimensions for students, courses and course instructors for different majors.

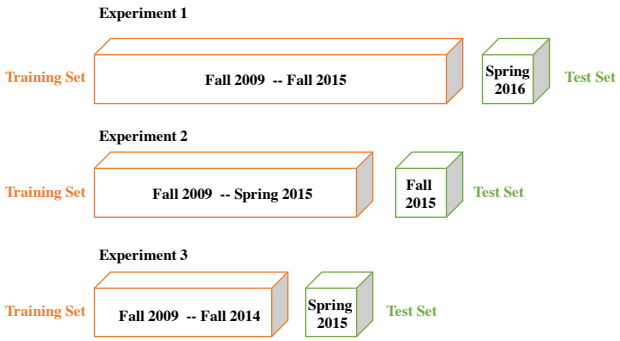

Figure 3: Experimental Protocols

C. Evaluation Metrics

In our dataset, a student's grade is a letter grade (i.e. A, A-, ..., F). These letter grades are first converted to numbers for both model training and testing. The predicted numerical grades are then converted into their closest letter grades (described in Section III-A). As in Polyzou *et. al.* [21], we use the **P**ercentage of **T**ick **A**ccuracy (PTA) to measure model performance. A tick is defined as the difference between two consecutive letter grades (e.g., C+ vs C or C vs C-; C+ and C- are 2-tick away). In particular, we compute the percentage of predicted grades with no error (or 0-ticks), within 1 tick and within 2 ticks, denoted by PTA_0, PTA_1 and PTA_2, respectively. Higher percentage of predictions within low ticks indicates better performance.

We also use Mean Absolute Error (MAE) to evaluate the predicted results in numbers. MAE is calculated as:

$$MAE = \frac{\sum_{s,c \in G_T} \left| g^l_{s,c} - \tilde{g}^l_{s,c} \right|}{|G_T|} \tag{5}$$

where $g^l_{s,c}$ and $\tilde{g}^l_{s,c}$ are the ground-truth grade and predicted grade for student s on course c taught by instructor l, respectively. G_T is the test set of (student, course, instructor, grade) tuples in the T-th term.

D. Baseline Methods

We compare our NCF methods with the following baseline methods.

1) Tensor Factorization: Tensor Factorization (TF) [1] has been successfully used in factorizing multi-way arrays and modeling relations among multiple types of elements. In our problem, since there are three elements, i.e., students, courses and course instructors, we use tensor factorization as one of our baseline methods to model their relations and make grade prediction.

We use $\mathcal{G}$ to represent a mode-3 student-course-instructor tensor, and each value in the tensor is the corresponding grade of a student in a course offered by an instructor; we use CANDECOMP/PARAFAC (CP) algorithm [2], [9], a very popular tensor decomposition algorithm, to decompose $\mathcal{G}$ into three matrices P, Q and R, where $P \in \mathbb{R}^{n \times k}$,

$Q \in \mathbb{R}^{m\times k}$, $R \in \mathbb{R}^{l\times k}$ are the matrices containing length-k latent factors for students, courses and course instructors in the same latent space, respectively. Here k is the number of latent factors. Thus, student s's grade on course c taught by instructor l is the combination of the Hadamard product [11] of the corresponding vectors in P, Q and R, that is,

$$\tilde{g}^l_{s,c} = \sum_{i=1}^{k} p_{s,i} q_{c,i} r_{l,i}. \tag{6}$$

To predict student's grade at t_{th} term, the loss function of TF is as follows:

$$\begin{aligned} L = & \sum_{g^l_{s,c}\in G^{t-1}} (g^l_{s,c} - \tilde{g}^l_{s,c})^2 \\ & + \alpha(\|P\|_2 + \|Q\|_2 + \|R\|_2) + \beta(|P|_1 + |Q|_1 + |R|_1) \end{aligned} \tag{7}$$

We add both l_2 and l_1 norms on matrices P, Q and R to prevent overfitting. We use stochastic gradient descent algorithm (SGD) to solve the optimization problem.

2) Non-negative Tensor Factorization: We further extend the above TF method and introduce non-negativity constraint on matrix P, Q and R [25]. This TF method with non-negativity constraint is referred to as non-negative tensor factorization and denoted as TF_{nn}.

3) Additive Latent Effect Models: We have developed Additive Latent Effect Models (ALE) for next-term grade prediction in our earlier work [22], and here we use ALE as one of the baseline methods with compare with. ALE considers student's academic levels, student's global effect and course instructors in addition to student and course knowledge to tackle grade prediction problem. The experimental results showed that ALE achieved significant improvement over several state-of-the-art baseline methods. The predicted grade from ALE is calculated as follows:

$$\tilde{g}^l_{s,c} = \mathbf{p}_s{}^{\mathsf{T}}(\mathbf{q}_c + \mathbf{r}_l), \tag{8}$$

where $\mathbf{p}_s$ ($\mathbf{p}_s \in \mathbb{R}^k$), $\mathbf{q}_c$ ($\mathbf{q}_c \in \mathbb{R}^k$) and $\mathbf{r}_l$ ($\mathbf{r}_l \in \mathbb{R}^k$) are the size-$k$ latent factors for student s, course c and course instructor l, respectively.

We also include non-negativity constraint on latent factor matrix P, Q and R on ALE model. This non-negative ALE model is denoted as ALE_{nn}. To predict student's grade at t_{th} term, the loss function of ALE is as follows

$$\begin{aligned} L = & \sum_{g^l_{s,c}\in G^{t-1}_s} (g^l_{s,c} - \tilde{g}^l_{s,c})^2 \\ & + \alpha(\|P\|_2 + \|Q\|_2 + \|R\|_2) + \beta(|P|_1 + |Q|_1 + |R|_1) \end{aligned} \tag{9}$$

Similarly, we add both l_2 and l_1 norms on matrices P, Q and R to prevent overfitting, and we use stochastic gradient descent algorithm (SGD) to solve the optimization problem.

VI. Results and Discussion

We now detail our experimental results and discuss the implications of these results.

A. Overall Performance

Table III shows the results in terms of the MAE, PTA_0, PTA_1, and PTA_2 metrics. We use three terms: Spring 2016, Fall 2015, and Spring 2015 as test sets, and we implement comprehensive experiments on every major. We use a grid search method to sweep over the embedding dimensions, and choose value 30 for student, course and course instructor embeddings dimensions. (parameter study on embedding dimensions will be presented later in Section VI-B)

We observe that both NCF and NCF_{nn} generally outperform the baselines across the different test sets. Specifically, for the Spring 2016 term, NCF outperforms the TF, TF_{nn}, ALE and ALE_{nn} baselines by 63.88%, 56.68%, 7.37% and 1.86% in terms of PTA_0, 43.85%, 51.19%, 11.15% and 9.72% in terms of PTA_1, and 17.45%, 18.19%, 10.41%, 9.41% in terms of PTA_2, respectively. However, we also notice that, for the Fall 2015 and Spring 2015 terms, the ALE baseline outperforms NCF and NCF_{nn} on several majors. This result is likely due to the fact that neural network-based methods can often overfit and perform poorly with insufficient training data; in our experiments, when we use the Fall 2015 and Spring 2015 terms as test sets, the amount of training data is significantly less than using the Spring 2016 term as test set (see Table II).

We note that when we use the Spring 2016 term as the test set, both NCF and NCF_{nn} significantly outperform every baseline. When we use the Fall 2015 and Spring 2015 terms as test sets, ALE and ALE_{nn} occasionally outperform NCF and NCF_{nn}. This observation agrees with that on the PTA metric. We also observe that when using the Spring 2016 term as test set, ALE_{nn} outperforms on the PTA_0 metric but underperforms NCF_{nn} on the PTA_2 metric, for the MATH, CHEM and IT majors. Meanwhile, NCF_{nn} always outperforms ALE_{nn} for all majors on the MAE metric. This observation shows that NCF_{nn} outperforms ALE_{nn} overall.

B. Effect of Embedding Dimensions

In order to get a deeper understanding on the impact of the embedding dimensions on model performance, we perform an experiment with different embedding dimensions for students, courses and course instructors. In total, we have eight different models, and we number each model $\text{NCF}_m (m = 0, 1, ..., 7)$; each model corresponds to a different set of embedding dimensions. NCF_0 corresponds to the NCF model in our previous experiments. Table IV shows the experimental results on using the Spring 2016 term as test set. We observe that, on different test sets, the best-performing model differs. For example, for the MATH major, NCF_2 performs best in terms of PTA while NCF_1 performs worst. However, for the CS major, NCF_1 performs best in terms of PTA while NCF_4 performs worst. This observation shows that the flexibility to choose different dimensions for student, course, and instructor embeddings is crucial, since the best performing models often have

Table III: Performance Comparison for All Methods

Test set: Spring 2016

Method	MATH				PSYC				CHEM			
	MAE(↓)	PTA_0 (↑)	PTA_1 (↑)	PTA_2 (↑)	MAE	PTA_0	PTA_1	PTA_2	MAE	PTA_0	PTA_1	PTA_2
TF	0.714	0.149	0.315	0.589	0.646	0.148	0.307	0.602	0.742	0.117	0.223	0.534
TF_{nn}	0.717	0.173	0.321	0.559	0.641	0.154	0.300	0.619	0.714	0.136	0.214	0.544
ALE	0.701	0.256	0.387	0.625	0.766	0.279	0.425	0.585	0.699	0.214	0.359	0.573
NCF	**0.617**	**0.262**	**0.435**	0.673	**0.508**	0.314	**0.506**	**0.737**	0.617	**0.194**	**0.408**	0.612
NCF_{nn}	0.618	0.238	0.429	**0.685**	0.515	**0.316**	0.487	0.726	**0.616**	0.184	**0.408**	**0.631**

Method	CS				IT				BIOL			
	MAE(↓)	PTA_0 (↑)	PTA_1 (↑)	PTA_2 (↑)	MAE	PTA_0	PTA_1	PTA_2	MAE	PTA_0	PTA_1	PTA_2
TF	0.790	0.126	0.249	0.483	0.685	0.172	0.349	0.588	0.631	0.236	0.395	0.636
TF_{nn}	0.783	0.133	0.253	0.502	0.687	0.207	0.344	0.586	0.685	0.171	0.299	0.591
ALE	0.719	0.186	0.348	0.584	0.652	0.230	0.389	0.649	0.651	0.236	0.397	0.626
NCF	0.691	**0.220**	**0.378**	0.616	0.604	**0.235**	**0.397**	0.691	**0.580**	**0.279**	**0.441**	**0.689**
NCF_{nn}	**0.690**	0.201	0.370	**0.619**	**0.602**	**0.235**	0.396	**0.698**	0.641	0.252	0.399	0.625

Test set: Fall 2015

Method	MATH				PSYC				CHEM			
	MAE(↓)	PTA_0 (↑)	PTA_1 (↑)	PTA_2 (↑)	MAE	PTA_0	PTA_1	PTA_2	MAE	PTA_0	PTA_1	PTA_2
TF	0.963	0.128	0.231	0.449	0.724	0.161	0.287	0.607	0.639	0.226	0.358	0.585
TF_{nn}	1.019	0.090	0.218	0.423	0.730	0.166	0.313	0.598	0.609	0.226	0.358	0.594
ALE	0.976	**0.244**	0.321	0.436	0.789	0.331	0.455	0.583	0.585	**0.274**	**0.462**	**0.660**
NCF	0.895	0.192	0.333	0.449	**0.561**	**0.366**	**0.550**	**0.728**	0.586	0.264	0.368	0.642
NCF_{nn}	**0.883**	0.218	**0.346**	**0.487**	0.601	0.322	0.497	0.706	**0.583**	0.245	0.368	0.642

Method	CS				IT				BIOL			
	MAE(↓)	PTA_0 (↑)	PTA_1 (↑)	PTA_2 (↑)	MAE	PTA_0	PTA_1	PTA_2	MAE	PTA_0	PTA_1	PTA_2
TF	0.749	0.116	0.237	0.533	0.656	0.185	0.328	0.604	0.698	0.174	0.302	0.553
TF_{nn}	0.716	0.139	0.290	0.533	0.617	0.180	0.355	0.663	0.729	0.134	0.249	0.529
ALE	**0.632**	**0.220**	**0.369**	**0.631**	0.645	**0.257**	0.412	0.645	**0.570**	**0.269**	0.408	**0.696**
NCF	0.672	0.139	0.283	0.604	**0.545**	**0.257**	**0.444**	**0.745**	**0.570**	0.254	**0.412**	0.664
NCF_{nn}	0.679	0.152	0.303	0.593	0.604	0.225	0.397	0.698	0.585	0.239	0.390	0.657

Test set: Spring 2015

Method	MATH				PSYC				CHEM			
	MAE(↓)	PTA_0 (↑)	PTA_1 (↑)	PTA_2 (↑)	MAE	PTA_0	PTA_1	PTA_2	MAE	PTA_0	PTA_1	PTA_2
TF	0.706	0.150	0.310	0.560	0.640	0.180	0.333	0.634	0.697	0.116	0.279	0.651
TF_{nn}	0.717	0.180	0.340	0.550	0.629	0.210	0.336	0.610	0.700	0.163	0.314	0.581
ALE	0.626	**0.370**	**0.520**	**0.670**	**0.518**	**0.310**	**0.513**	**0.745**	0.802	0.186	**0.372**	0.558
NCF	0.596	0.280	0.430	0.640	0.543	0.267	0.455	0.711	0.672	**0.198**	0.302	**0.628**
NCF_{nn}	**0.595**	0.270	0.440	0.660	0.538	0.259	0.461	0.713	**0.660**	**0.198**	0.291	0.616

Method	CS				IT				BIOL			
	MAE(↓)	PTA_0 (↑)	PTA_1 (↑)	PTA_2 (↑)	MAE	PTA_0	PTA_1	PTA_2	MAE	PTA_0	PTA_1	PTA_2
TF	0.733	0.140	0.278	0.538	0.639	0.215	0.342	0.624	0.663	0.184	0.321	0.591
TF_{nn}	0.711	0.153	0.300	0.552	0.634	0.217	0.357	0.618	0.658	0.182	0.324	0.581
ALE	0.701	**0.218**	**0.378**	0.613	0.643	0.259	0.428	0.660	0.600	**0.255**	**0.395**	**0.669**
NCF	**0.657**	0.211	0.363	0.620	0.574	**0.276**	**0.456**	0.703	0.600	0.210	0.371	0.645
NCF_{nn}	0.658	0.194	0.344	**0.622**	**0.570**	0.268	**0.456**	**0.719**	**0.599**	0.220	0.373	0.648

"↓" indicates the lower the better, and ↑" indicates the higher the better. The best performing methods are highlighted with bold.

Table IV: Performance Comparison for Different Embeddings Dimensions

Model	# StdEm	# CrsEm	# InstrEm	MATH			PSYC		
				PTA_0	PTA_1	PTA_2	PTA_0	PTA_1	PTA_2
NCF_0	30	30	30	0.262	0.435	0.673	0.314	**0.506**	0.737
NCF_1	15	30	30	0.239	0.387	0.598	0.306	0.487	0.724
NCF_2	30	15	30	**0.299**	**0.461**	**0.705**	0.300	0.483	0.733
NCF_3	30	30	15	0.294	0.453	0.694	**0.319**	0.498	0.733
NCF_4	15	15	30	0.272	0.418	0.677	0.306	0.478	0.729
NCF_5	30	15	15	0.246	0.390	0.603	**0.319**	0.501	**0.740**
NCF_6	15	30	15	0.275	0.421	0.676	0.311	0.485	0.729
NCF_7	15	15	15	0.250	0.452	0.696	0.307	0.477	0.732

Model	# StdEm	# CrsEm	# InstrEm	CHEM			CS		
				PTA_0	PTA_1	PTA_2	PTA_0	PTA_1	PTA_2
NCF_0	30	30	30	0.194	0.408	0.612	0.220	0.378	0.616
NCF_1	15	30	30	0.184	0.398	0.602	**0.227**	0.391	**0.635**
NCF_2	30	15	30	**0.243**	0.379	0.612	**0.227**	0.382	0.619
NCF_3	30	30	15	0.194	0.417	**0.621**	0.214	0.379	0.625
NCF_4	15	15	30	0.184	0.408	0.592	0.211	0.366	0.628
NCF_5	30	15	15	0.184	**0.437**	0.612	0.223	**0.394**	0.618
NCF_6	15	30	15	0.175	0.408	0.612	0.224	0.359	0.634
NCF_7	15	15	15	**0.243**	0.398	**0.621**	0.209	0.381	0.615

Model	# StdEm	# CrsEm	# InstrEm	IT			BIOL		
				PTA_0	PTA_1	PTA_2	PTA_0	PTA_1	PTA_2
NCF_0	30	30	30	0.235	0.397	0.691	0.279	0.441	0.689
NCF_1	15	30	30	0.237	0.397	0.691	0.215	0.372	0.602
NCF_2	30	15	30	0.229	0.397	0.686	**0.299**	**0.461**	**0.705**
NCF_3	30	30	15	0.220	0.407	**0.701**	0.294	0.453	0.694
NCF_4	15	15	30	0.227	0.392	0.683	0.272	0.418	0.677
NCF_5	30	15	15	0.239	**0.409**	0.694	0.210	0.378	0.614
NCF_6	15	30	15	**0.244**	**0.409**	0.693	0.275	0.421	0.676
NCF_7	15	15	15	0.230	0.394	0.699	0.264	0.427	0.677

StdEm, # CrsEm and # InstrEm indicate the dimensions of student embeddings, course embeddings and course instructor embeddings, respectively.

different values for the dimension of these embeddings. Therefore, this flexibility enables neural networks-based models to outperform matrix factorization-based models, which use the same dimension for every embedding.

C. Effect of Non-Negativity Constraint

For simplicity of exposition, we only show results on the PTA_2 metric for all methods and under all experimental protocols in Fig. 4, since the results on the other metrics are similar. We observe that for three head-to-head comparisons, i.e., TF versus TF_{nn}, ALE versus ALE_{nn} and NCF versus NCF_{nn}, in most circumstances, each pair of methods show similar results on different majors, such as the CS major. However, in some cases, adding the non-negativity constraint can lead to significant changes in model performance, e.g., ALE and ALE_{nn} on PSYC major in the experiments with Spring 2015 as test set. Further observing Fig 4, we notice that for the experiment with Spring 2015 as test set, NCF_{nn} is more likely to outperform NCF on different majors than the experiments with Spring 2016 and Fall 2015 as test sets, respectively. This shows that with insufficient training samples, non-negativity constraint can help NCF_{nn} model student, course and course instructors better than NCF, and finally gain better grade prediction results.

VII. Conclusion

In this work, we develop a new deep learning inspired neural collaborative filtering approach for solving the next-term grade prediction problem. We consider three elements as input to the NCF model i.e., student, course and course

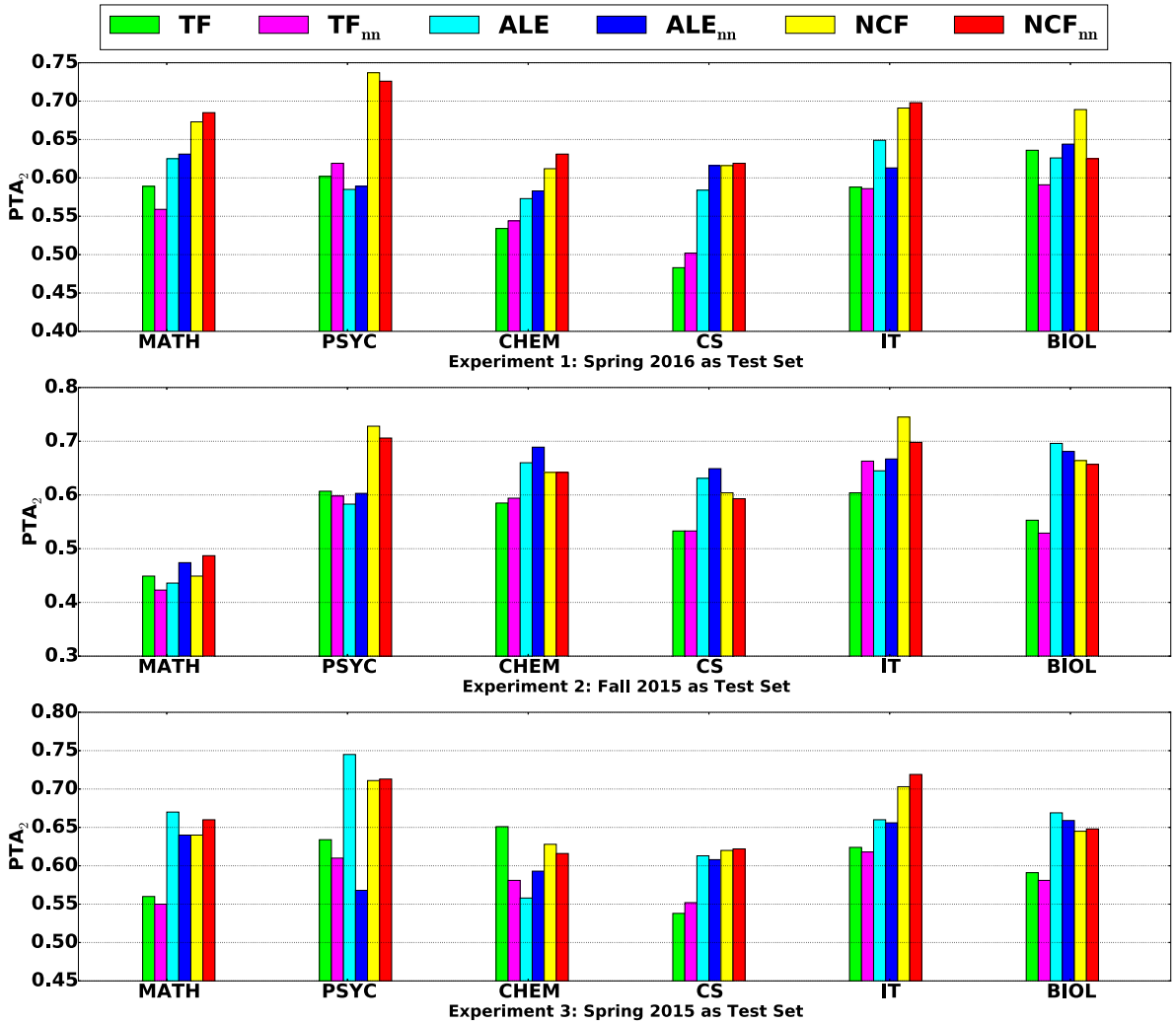

Figure 4: Analysis on the Effect of Non-Negativity Constraint

instructor. Furthermore, the learned embeddings of these three input elements can be considered as the "hidden" factors within the classic latent factor model in collaborative filtering techniques. For proper analysis of the model, we also add non-negativity constraints on the embeddings by adding Rectified Linear Units (ReLU) on the embedding layer. The experimental results demonstrate that both NCF and NCF$_{nn}$ significantly outperform the baselines on grade prediction problem over various test sets. In addition, we analyze the model performance with different embeddings dimensions for student, course and course instructor, respectively. The results show that NCF provides flexibility in that, different from classic latent factor models in collaborative filtering techniques, different elements can have various dimensions and achieves better results than the one with the same embedding dimensions for all the elements. Finally, we provide in-depth analysis on the effect of non-negativity constraint. We have found that with insufficient training samples, non-negativity constraint can help NCF$_{nn}$ model student, course and course instructors better than NCF, and finally gain better grade prediction results.

REFERENCES

[1] Evrim Acar, Daniel M Dunlavy, Tamara G Kolda, and Morten Mørup. Scalable tensor factorizations for incomplete data. *Chemometrics and Intelligent Laboratory Systems*, 106(1):41–56, 2011.

[2] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[3] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 191–198. ACM, 2016.

[4] Asmaa Elbadrawy and George Karypis. Domain-aware grade prediction and top-n course recommendation. *Boston, MA, Sep*, 2016.

[5] Asmaa Elbadrawy, Agoritsa Polyzou, Zhiyun Ren, Mackenzie Sweeney, George Karypis, and Huzefa Rangwala. Predicting student performance using personalized analytics. *Computer*, 49(4):61–69, 2016.

[6] Asmaa Elbadrawy, Scott Studham, and George Karypis. Personalized multi-regression models for predicting students performance in course activities. *UMN CS 14-011*, 2014.

[7] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *Proceedings of the 24th International Conference on World Wide Web*, pages 278–288. International World Wide Web Conferences Steering Committee, 2015.

[8] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.

[9] Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

[10] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 173–182. International World Wide Web Conferences Steering Committee, 2017.

[11] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 1990.

[12] Chein-Shung Hwang and Yi-Ching Su. Unified clustering locality preserving matrix factorization for student performance prediction. *IAENG Int. J. Comput. Sci*, 42(3):245–253, 2015.

[13] Bin Ju, Yuntao Qian, Minchao Ye, Rong Ni, and Chenxi Zhu. Using dynamic multi-task non-negative matrix factorization to detect the evolution of user preferences in collaborative filtering. *PloS one*, 10(8):e0135090, 2015.

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] Severin Klingler, Rafael Wampfler, Tanja Käser, Barbara Solenthaler, and Markus Gross. Efficient feature embeddings for student classification with variational auto-encoders.

[16] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[17] Yehuda Koren, Robert Bell, Chris Volinsky, et al. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[18] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[19] Michelle Parker. Advising for retention and graduation. 2015.

[20] Štefan Pero and Tomáš Horváth. Comparison of collaborative-filtering techniques for small-scale student performance prediction task. In *Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering*, pages 111–116. Springer, 2015.

[21] Agoritsa Polyzou and George Karypis. Grade prediction with models specific to students and courses. *International Journal of Data Science and Analytics*, pages 1–13, 2016.

[22] Zhiyun Ren, Xia Ning, and Huzefa Rangwala. Ale: Additive latent effect models for grade prediction. *arXiv preprint arXiv:1801.05535*, 2018.

[23] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B Kantor. Recommender systems handbook., 2011.

[24] Arjun Sharma, Arijit Biswas, Ankit Gandhi, Sonal Patil, and Om Deshmukh. Livelinet: A multimodal deep recurrent neural network to predict liveliness in educational videos. In *EDM*, pages 215–222, 2016.

[25] Amnon Shashua and Tamir Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning*, pages 792–799. ACM, 2005.

[26] Jill M Simons. *A National Study of Student Early Alert Models at Four-Year Institutions of Higher Education.* ERIC, 2011.

[27] Mack Sweeney, Jaime Lester, and Huzefa Rangwala. Next-term student grade prediction. In *Big Data (Big Data), 2015 IEEE International Conference on*, pages 970–975. IEEE, 2015.

[28] Mack Sweeney, Huzefa Rangwala, Jaime Lester, and Aditya Johri. Next-term student performance prediction: A recommender systems approach. *arXiv preprint arXiv:1604.01840*, 2016.

[29] Nguyen Thai-Nghe, Lucas Drumond, Artus Krohn-Grimberghe, and Lars Schmidt-Thieme. Recommender system for predicting student performance. *Procedia Computer Science*, 1(2):2811–2819, 2010.

[30] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1235–1244. ACM, 2015.

[31] Xiaolu Xiong, Siyuan Zhao, Eric Van Inwegen, and Joseph Beck. Going deeper with deep knowledge tracing. In *EDM*, pages 545–550, 2016.

[32] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. Bridging collaborative filtering and semi-supervised learning: A neural approach for poi recommendation. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1245–1254. ACM, 2017.