MACHINE LEARNING FOR STUDENT MODELING AND FORECASTING

by

Qian Hu
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Computer Science

Committee:

_____	Dr. Huzefa Rangwala, Dissertation Director
_____	Dr. Daniel Barbará, Committee Member
_____	Dr. Vadim Sokolov, Committee Member
_____	Dr. Thomas LaToza, Committee Member
_____	Dr. Huzefa Rangwala, Department Chair
_____	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date: _____ Spring Semester 2020
George Mason University
Fairfax, VA

Machine Learning for Student Modeling and Forecasting

A dissertation submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy at George Mason University

By

Qian Hu
Bachelor of Engineering
Nanjing University of Aeronautics and Astronautics - Nanjing, China, 2014

Director: Dr. Huzefa Rangwala, Professor
Department of Computer Science

Spring Semester 2020
George Mason University
Fairfax, VA

Dedication

Dedicated to my parents.

Acknowledgments

First and foremost, I would like to express my great gratitude to my advisor and friend, Huzefa Rangwala. During my PhD journey, he has provided me with enormous amount of help and guidance that greatly influenced and shaped the work I have done. I would like to thank my committee members Daniel Barbará, Thomas LaToza and Vadim Sokolov for their time in reviewing. Particularly, I want to thank Vadim Sokolov for introducing me into Bayesian deep learning. I would also like to thank my collaborators Agoritsa Polyzou and George Karypis. It is a great pleasure to work with them. Finally, I would like to thank lab members at the Data Mining Lab for useful discussion and conversations.

This research was supported by National Science Foundation Grant #1447489.

Table of Contents

List of Tables

List of Figures

Abstract

MACHINE LEARNING FOR STUDENT MODELING AND FORECASTING

Qian Hu, PhD

George Mason University, 2020

Dissertation Director: Dr. Huzefa Rangwala

Higher educational institutions face major challenges including timely graduation and retention of enrolled students. The National Center for Education Statistics (NCES) reports that the six-year graduation rate for first-time and full-time undergraduates is around 60%; the retention rate among first-time and full-time degree-seeking students is around 80%. These alarming statistics require higher educational institutions to take actions to improve their effectiveness and efficiency at educating students. Educational data mining technologies for academic trajectory and degree planning, course recommender systems, early warning and advising seek to improve student success. The foundation of these systems is student modeling and forecasting.

However, developing appropriate and accurate predictive models for modeling students is a non-trivial problem due to several challenges. The first challenge is that a student's learning is influenced by many factors such as motivation, affect and identify. It is further compounded by the fact that learning is a reflection of cognition which is not a simple process. Students can also choose to take courses in different sequences at different pace. The second challenge is that degree programs exhibit complex knowledge dependence between courses. When it comes to decision making, machine learning has its shortcomings in terms of predictive reliability and interpretability. A reliable model is able to express its prediction confidence so that human decision-makers can know when the predictions are trustworthy. Interpretable models can provide explanations for predictions and

decision-makers can use the explanations to guide their decisions. Recently, several concerns have emerged about the fairness of using machine learning models. A biased predictive model may negatively influence subgroups of the larger population. For example, in the educational setting, models unfairly predicting a particular group of students to be at a higher risk of failure can discourage them from pursuing their degree pathway.

In this thesis, we develop novel and accurate machine learning models for student modeling and forecasting. Specifically, we develop sequential and graph machine learning models to model students' learning processes and predict their academic performance. Towards informed decision making, we develop Bayesian deep learning models to quantify uncertainty. We also propose a metric-free individual fairness formalization and develop two fair machine learning models using neural classifiers and gradient contextual bandits for mitigating unfairness in these predictive models.

Chapter 1: Introduction

1.1 Introduction

The average six-year graduation rate for undergraduate programs in the United States has been around 59% for over a decade [1]. More than half of the graduating students take six years to finish four-year programs. The additional time required by students and low graduation rates has high human, monetary and societal costs with regards to workforce training and economic growth. Lack of proper academic preparation and planning are some of the main reasons that lead to student failure in higher education [2]. Technologies from educational data mining have been identified as promising for improving students learning. In this research, we focus on developing foundamental algorithms for student modeling and forecasting. These algorithms can be used to build applications for aiding students such as educational early warning systems, degree planners and course recommender systems, to name a few.

1.1.1 Student Modeling

Educational data mining (EDM) is a discipline concerned with developing machine learning and data mining methods utilizing student-related data to better understand students and aid their learning [3]. Many approaches have been developed for tasks including student performance prediction [4, 5], affect detection [6, 7], graduation prediction [8, 9], etc. All these tasks are built upon a fundamental task which is student modeling. Accurately modeling student is challenging due to several reasons such as i) the complexity and flexibility of student's learning process ii) the need for reliable and interpretable predictions for decision making.

Complexity and Flexibility

Student's learning is complex as it is affected by many factors such as motivation, learning habits, attention, environment and pedagogy, etc [10], Meanwhile, it is flexible in the sense that different students choose different courses to fulfill the needs of their degree requirements and they can follow different orders to take those courses. In addition, students take courses to accumulate knowledge and knowledge requirement of a course comes from different prior courses. Knowledge dependence between courses in a program is connected in a complex way. The combination of these factors makes student modeling a highly complex task.

Reliability and Interpretability

The ultimate goal of student modeling is for decision making such as student intervention. For decision making, predictive reliability and interpretability are important for decision makers to trust and make informed decisions [11]. A reliable model should exhibit high uncertainty when the prediction is not trustworthy so that decision makers can take over and decide based on their own judgement. A model provides explanation for their predictions can help decision makers know the reasons behind them and make informed decisions and thus provide accurate feedback to students. For example, if a model using student's grades in prior courses as features for predicting at-risk students is able to explain which prior courses accounting for students failure, we can provide accurate feedback to students and let them improve their learning on those prior courses to succeed in the target course. We will discuss methods we developed to achieve reliable and interpretable predictions.

1.1.2 Fairness

Recent progress in machine learning has sparked concerns about its fairness and equality. Many studies have shown evidence of bias in machine learning models [12–14]. Machine learning models are trained on data. If data is biased, the model trained on biased data is discriminative [15]. Discriminative machine learning models can lead to ethical and legal consequences for stakeholders. In educational setting, biased predictions can discourage students and undermine their learning

outcome. We thoroughly analyzed a real-world educational dataset collected at George Mason University and found clear bias embedded in the data with respect to gender and race. Specifically, we found that the overall female GPA is higher than male GPA (3.15 vs. 2.86). The GPA of African-American is lower than non-African-American (2.86 vs. 3.03). To mitigate bias, we developed a novel fairness concept and proposed two fair models based on classifiers and gradient contextual bandits.

1.2 Problem Statement

The ultimate goal of student modeling is for student forecasting. In this research, we focus on two predictive tasks i) student performance prediction and ii) identifying at-risk students. We treat student performance prediction as a regression task. Namely, given a student and his historical data such as grades in prior courses, we want to predict his/her grade in a future target course. Grade prediction can be useful for detecting at-risk students. A study by Polyzou et al. [16] shows that for at-risk student detection, a dedicated classifier is more desirable. Based on this observation, we treat the task of at-risk student detection as a classification task. That is given a student with features extracted from his historical data, we want to predict whether he/she will be at-risk of failing a target course. Besides the accuracy of the classifiers, we also focus on fairness of the predictive models.

1.3 Organizations and contributions

We developed methods to resolve the aforementioned challenges in educational data mining and test the methods on the task of student performance prediction and at-risk student detection. They are summarized as following

- (Chapter 2) In this chapter, we discuss background of student modeling and prediction and the fundamental knowledge that our methods are based upon.
- (Chapter 3) To improve the accuracy of student modeling, we developed a hybrid model that utilizes both student's features and grades of prior courses. This model solves data missing issue in traditional course-specific models.

- (Chapter 4) Students take courses sequentially. To account for temporal dynamics within student's course taking process, we proposed course-specific Markovian models for student modeling and grade prediction.

- (Chapter 5) Traditional models for student modeling and performance prediction are simple and linear models. However, student's knowledge acquiring process is highly complex and shallow linear models are not able to fully model students. To better model students, we proposed deep learning models for student modeling. We also proposed Bayesian deep learning models for predictive reliability and developed a simple method for explaining the models' prediction.

- (Chapter 6) Undergraduate degree programs exhibit complex knowledge dependence and we found the knowledge dependence are structured as graphs. To model knowledge dependence between courses, we proposed graph convolutional networks based on attention for student modeling.

- (Chapter 7) Finally, to mitigate machine learning bias, we first proposed a metric-free individual fairness formalization and proposed fair models to implement this formalization.

Chapter 2: Background

The application of analytics to improve educational quality can be seen in many areas related to modeling of learners [17], predicting and advising learners [3], automated content enhancement [18], knowledge tracing [19, 20] and course/topic recommendations to students [21]. Among them, student's academic performance prediction has attracted much attention, as it underlies applications to several AI-based decision making systems including educational early warning systems, degree planning and academic trajectory planning [3]. In this part, we review related works.

2.1 Student Prediction

Several machine learning algorithms have been applied to tackle the student performance prediction problem [22, 23]. Al-Barak et al. applied decision trees for grade prediction by using students' transcript data [24]. Umair et al. used Support Vector Machines (SVMs) to select key training instances for grade prediction [25]. Recommender systems based methods including collaborative filtering [26], matrix factorization [27] and factorization machines [4] have been proposed for grade prediction. These approaches use a one-size-fits-all framework for training the model and prediction. Polyzou et al. proposed a personalized model that is specific to each course and student [28]. Student-course enrollment patterns have grouping structures which result in missing not at random patterns of student grade data. Leveraging this, Elbadrawy et al. proposed a domain-aware grade prediction algorithm for student's performance prediction and course recommendation [21]. Since students accumulate knowledge by taking courses sequentially within the academic programs, it is assumed that the knowledge state of the students is evolving. Ren et al. proposed a temporal course-wise influence model which incorporates the influence of prior courses in a sequential way, however, up to two terms [29]. Course-specific regression models cannot correctly capture students' knowledge state when the same knowledge can be acquired by taking different subsets of courses.

To solve this problem, Morsy et al. [30] developed Cumulative Knowledge-based Regression Model (CKRM), which represents the knowledge state of students in knowledge component vectors.

2.2 Temporal Dynamic Models

Hidden Markov Models (HMM) were first introduced and extensively studied by Baum et. al. [31–33]. These models have found success in several fields including speech recognition, gene prediction and time series analysis. The application of HMM to education was first proposed by Corbett et al. [20] to model the acquisition of procedural knowledge in intelligent tutoring system as Bayesian Knowledge Tracing (BKT). In this model, the hidden states represent whether a student masters a skill or not and the observations are the students' answers to a series of questions. Extending BKT, several models have been proposed such as the individualized BKT [19] to improve the prediction performance. HMM have also been applied for predicting dropouts in Massively Open Online Courses (MOOCs). Balakrishnan et al. [34] used a novel Input-Output HMM to predict student retention and inferred the general behavior patterns between the students completing courses and those dropping at different points in a term. Geigle et al. proposed a two-layer Hidden Markov Model to model student behavior in MOOCs and found that the features extracted from the two-layer Hidden Markov Model correlated with educational outcomes [35]. Such modeling efforts involved extracting server logs of student interaction from the online systems with the objectives of identifying students at-risk of dropping out. The observation layer of the two-layer HMM is used to model the sequence of students' interactions with the systems [35].

Hidden Semi-Markov Models (HSMMs) were first proposed in the area of speech recognition [36]. The key advantage of these models is that they can explicitly model the distribution of hidden state duration. Since then, HSMMs have been applied to areas including computer vision, DNA analysis, protein structure prediction and financial time series [37]. To the best of our knowledge, HSMM models have not been applied to educational datasets and for the problem of next-term grade prediction.

2.3 Deep Learning for Educational Data

The success of deep learning comes from its hierarchical architecture introduced by stacking layers. Each layer processes some part of the information and passes it to the next layer. Each layer solves a part of the task before passing it on to the next, until the last layer provides the output. Hierarchical architecture enables deep learning to learn very complex patterns within data. Student's learning process is highly complex that shallow linear learners are not able to capture. To better model student's learning process, deep learning models have been proposed in educational data mining research community to model student's learning habits and predict performance. Livieris et al. developed a neural network based classifier to predict whether a student will have poor performance in a Math course [38]. Gedeon et al. trained a feedforward neural network to predict a student's final grade in a computer science course using data from teaching sessions and provided interpretability of the prediction results by generating a set of rules [39]. Yang et al. designed a time series neural network using a student's clickstream data while watching video lectures in massive open online courses (MOOCs) [40]. Okubo et al. proposed a recurrent neural network classifier to predict a student's grade by using data from various logs of learning activities [41]. For modeling student's learning process within Intelligent Tutoring Systems (ITS), Piech et al. proposed using deep knowledge tracing [42]. Most of the proposed neural network models were developed for in-class prediction or for intelligent tutoring systems that model student learning in a single course. The DKT models are similar to our proposed LSTM model; however, the DKT models only incorporate one response each time step. Our proposed LSTM model is more flexible and can incorporate several (responses) grades of prior courses taken together in the same semester.

2.4 Predictive Uncertainty

Deep learning models have achieved state-of-the-art performance in many areas due to their abilities to model complex patterns [43]. However, general deep learning models cannot represent uncertainty, which is critical for decision-making. Bayesian models have the advantage of providing principled uncertainty estimation. Therefore, combining Bayesian approaches with deep learning

models is a way to obtain benefits from these two perspectives.

Bayesian deep learning models place a prior distribution over model parameters; the model is updated by Bayes' rule with observed data. The posterior distribution of the model parameters is the learned model. Due to possible non-linear activation functions that can be applied to neurons, exact model posterior is not available. Approximate inference methods are used for model training, such as variational inference [44]. However, these methods have a high computational cost and are hard to scale in practice.

Recently, Monte Carlo (MC) dropout has been proposed by Gal et al. [11], which is efficient for uncertainty estimation and requires no change in the designed model architecture.

2.5 Fairness

Machine learning models have been deployed in many areas to help guide human decision makers. However, recent studies show that machine learning models are prone to make biased and discriminative predictions. Biased predictions can hurt the interest and benefits of minority groups and lead to ethical and legal conflicts. In order to resolve this issue, machine learning community have proposed fairness formalizaitons and fair machine learning models.

2.5.1 Fairness formalizations

Statistical parity has been proposed which requires that a predictor predicts a particular outcome for individuals across groups with nearly equal probability [45,46]. Hardt et al. [47] proposed equalized odds and equal opportunity. This formalization requires that conditioned on the true outcomes, the predicted outcomes should be independent of the sensitive attributes. Based upon causal inference, Kusner et al. [48] proposed counterfactual fairness which states that the predictive outcome for an individual stays the same if his/her sensitive attribute is changed to its counterfactual value. Dwork et al. [49] proposed individual fairness which requires that similar individuals should be treated similarly.

2.5.2 Fair algorithms

Many fair algorithms have been proposed. Rawlsian fairness has been proposed by Joseph et al. [50] which is based on the notion of fairness by John Rawls that a worse candidate should never be favored by a better one. The proposed algorithm is based on contextual bandits. The action value of the contextual bandits is treated as the qualification of an individual. Individuals are differentiated by their qualification. Zemel et al. [51] proposed to learn a fair representation that removes sensitive information and assume that classifiers built upon the representation are fair. Following the idea of learning fair representation, Edwards et al. [52] proposed to remove sensitive information by using adversarial learning.

Chapter 3: Content Features Enriched Course-Specific Hybrid model for Grade Prediction

Traditional grade prediction methods rely on one-size-fits-all models, namely, training a model to predict a student's grades of all next-term courses [53]. As different courses have different knowledge structure and requirements, one major issue of this kind of model is that it ignores unique features of different courses. To overcome this issue, course-specific models have been proposed to predict student's next-term grades by using grades of prior courses, which better addresses pertinent challenges associated with the reliable estimation of the low-rank models [5]. However, course-specific models that use the grades of prior courses can only capture the information of student's knowledge evolution. Course-specific models also suffer from inaccurate prediction if the degree program is flexible (i.e., has several electives). In addition, there are some other factors that can influence student's grades, such as his/her academic level when taking a certain course, instructor's teaching quality and courses' difficulty. To solve this problem we incorporate content features, which can capture diverse information about students, courses and instructors. Based on course-specific models, we present a model which not only uses the grades of prior courses but also different kinds of content features.

We evaluated our proposed method on a dataset from George Mason University (GMU) collected from Fall 2009 to Spring 2016 and on a dataset from University of Minnesota (UMN) collected from Fall 2003 to Spring 2014. The results showed that our proposed method outperformed competing methods to some degree. Another finding was that when the prior-course information was sparse, the included content features were more likely to help. However, as the availability of content features in the two universities is different, namely in GMU we have more informative content features, for majors with flexible degree programs in GMU, the course-specific model with content features achieves the best performance; for majors with flexible degree programs in UMN,

the proposed course-specific model with grades of prior courses and content features performs better. This suggests that the availability of the content features can influence the performance of the proposed model.

3.1 Problem Formulation and Notation

Formally, we assume that we have records of n students and m courses, comprising a $n \times m$ sparse grade matrix $\mathbf{G}$, where $g_{s,c} \in [0 - 4]$ is the grade a student s earned in course c. The objective of next-term grade prediction problem is to estimate the grade $\hat{g}_{s,c}$, a student s will achieve in course c in the next term. Besides the grade matrix $\mathbf{G}$, we have information that can be associated with the student (e.g., academic level, previous GPA, major) and course offering (e.g., discipline, course level, prior courses frequently taken, instructor, etc) that can be combined to extract a feature vector per dyad. We denote this feature vector as $\mathbf{x}$ of p dimensions. As a convention, bold uppercase letters are used to represent matrices (e.g., $\mathbf{X}$) and bold lowercase letters represents vectors (e.g., $\mathbf{x}$).

3.2 Methods

3.2.1 Course-Specific Regression with Prior Courses

Polyzou et.al. [5] motivate the use of course-specific regression models that leverage the sequential structure of undergraduate degree programs. These regression models assume that the performance of a student in a future course is strongly correlated with past performance on a subset of courses related to the degree program taken earlier. Specifically, this regression model estimates the grades for a future class as a sparse linear combination of grades obtained on prior courses. For a course c the grades that students obtained on courses taken prior to c are extracted from the grade matrix $\mathbf{G}$, and denoted by $\mathbf{G}_c^{pr}$. Each row of this matrix corresponds to students that have taken the course c. Assume that n_c students have taken the course c so far and m_c represents the union set of courses taken by students prior to c, then the dimensions of $\mathbf{G}_c^{pr}$ is $n_c \times m_c$. $\mathbf{g}_{:,c}$ is the vector representing the grades that students obtained for course c. We learn the parameters of this Course-Specific Regression (CSR) model by solving the least square regression problem enforcing $\ell 1$ and $\ell 2$ norms.

The optimization problem is given below:

$$min \underbrace{||\mathbb{1}w_{c0} + \mathbf{G}_c^{pr}\mathbf{w}_c^{pr} - \mathbf{g}_{:,c}||_2^2}_{\text{loss}} + \underbrace{\lambda_1||\mathbf{w}_c^{pr}||_2^2}_{\ell 2} + \underbrace{\lambda_2||\mathbf{w}_c^{pr}||_1}_{\ell 1} \tag{3.1}$$

where $\mathbb{1}$ is a vector of ones of dimension n_c, $\mathbf{w}_c^{pr} \in R^{m_c}$ denotes the weight vectors associated with each course c and $w_{c,0}$ is the bias term. The $\ell 1$ norm promotes sparsity and $\ell 2$ norm prevents overfitting.

Having learned the weight vectors and bias terms, the grade estimate for a student s enrolling in course c is given by:

$$\hat{g}_{s,c} = w_{c0} + \mathbf{x}_{s,c}^T\mathbf{w}_c^{pr} \tag{3.2}$$

where $\mathbf{x}_{s,c} \in \mathbb{R}_c^m$ is a feature vector representing the grades on prior courses that the student has taken so far. We denote this Course-Specific Regression model with Prior Courses as *CSR$_{PC}$*.

In this approach, prior to estimating the model using equation 3.1, we row-centered each row of matrix $\mathbf{G}_c^{pr}$ and $\mathbf{g}_{:,c}$, which is done by subtracting the GPA of corresponding students from the non-zero entries in each row of $\mathbf{G}_c^{pr}$ and $\mathbf{g}_{:,c}$ [5]. We found that row-centering gives better performance by mitigating the negative influence of missing grades from prior courses.

3.2.2 Course-Specific Regression with Content Features

The CSR$_{\text{PC}}$ model described above is able to provide accurate estimates of student performance in a course provided that the students taking that course has commonly taken sufficient number of prior courses. We seek to extract key features associated with students and courses and incorporate them within the prediction formulation. Based on course-specific idea, instead of training one global model for all the courses as done in existing work [4], we propose to train independent course-specific regression models with content features. We refer to this model by CSR$_{\text{CF}}$. In terms of formulation, the proposed CSR$_{\text{CF}}$ is similar to CSR$_{\text{PC}}$ except that the feature vector is a composite of student, course and instructor-related features as described below.

We denote the weight vector learned by this formulation as $\mathbf{w}_c^f$ and the feature vectors $\mathbf{x}_{s,c} \in \mathbb{R}^p$

where p is the total number of features. The predicted grade estimate is then given by:

$$\hat{g}_{s,c} = w_{c0} + \mathbf{x}_{s,c}^T \mathbf{w}_c^f \tag{3.3}$$

The CSR_{CF} model is estimated in a similar manner as CSR_{PC} and given by:

$$min \underbrace{||\mathbb{1}w_{c0} + \mathbf{X}_c^f \mathbf{w}_c^f - \mathbf{g}_{:,c}||_2^2}_{\text{loss}} + \underbrace{\lambda_1 ||\mathbf{w}_c^f||_2^2}_{\ell 2} + \underbrace{\lambda_2 ||\mathbf{w}_c^f||_1}_{\ell 1} \tag{3.4}$$

where $\mathbf{X}_c^f$ is a matrix of stacked feature vectors from the different students who have taken the course c in the past. Each row of this matrix is a feature vector for a student enrolled in the course c.

Content features for GMU

1. Student Features. Student-related features include their demographic data, such as their age, race, gender, high school GPA and so on. For each term, we have the GPA of the student from the previous term and the accumulative GPA as of last term. As students might take courses from other departments which has less influence than those from their own departments, we can extract GPA of courses only from their own departments. When taking a course, different students might come from different academic level, therefore, it might be beneficial to incorporate their academic level into the model.
2. Course Features. The features relating to a course include its discipline, the credit hours and course level (e.g. 100, 200, 300, 400-level). As the difficulty of a course can influence the performance of the students, we include the course difficulty information into the model. We use the GPA of the course from last term to represent the difficulty of the course.
3. Instructor Features. As the factors from instructors can also influence the performance of the students, we extract content features about the instructors which include rank, tenure status and the GPA of the courses he has taught.

Content features for UMN

1. Student Features. Same as in GMU apart from the features related to demographic data. Considering a specific term for which a student has taken a course, we extracted their GPA of the previous term, the accumulative GPA as of last term, the GPA over only courses from their own departments, as well as, the students' academic level.
2. Course Features. Same as the ones extracted for GMU.
3. Instructor Features. No instructor features are available.

We one-hot-encoded categorical features in $\mathbf{X}_c^f$ and standardized the continuous features.

3.2.3 Hybrid Model

We also combine the feature vectors $\mathbf{X}_c^f$ and $\mathbf{G}_c^{pr}$ obtained from the student-course content and prior grades and learn weight vectors per course, respectively. We refer to this hybrid model as CSR_{HY} and learn a course-specific regression model as discussed above.

3.2.4 Baseline Methods

In the experiments, we compare the proposed methods with the following baseline approaches.

1. BiasOnly (BO): BiasOnly method only takes into consideration student's bias, course's bias and global bias which are estimated using Equation 4.7.

$$\hat{g}_{s,c} = b_0 + b_s + b_c \tag{3.5}$$

   where b_0, b_s and b_c are the global bias, student bias and course bias respectively.

2. Matrix Factorization (MF): The use of MF for grade prediction is based on the assumption that the students and courses' knowledge space can be jointly represented in low-dimensional latent feature space [5]. Each component in the latent feature space corresponds to knowledge components. The grade of student s in a future course c is estimated as:

$$\hat{g}_{s,c} = b_0 + b_s + b_c + \boldsymbol{p}_s^T \boldsymbol{q}_c \tag{3.6}$$

where b_0, b_s and b_c are the global bias, student bias and course bias respectively and $\boldsymbol{p}_s$, $\boldsymbol{q}_c$ are the latent vectors representing student s and course c.

3. Course-specific Matrix Factorization (CS_{MF}): CS_{MF} is similar to MF except that the grade matrix $\mathbf{G}_c$ for CS_{MF} only includes the grades of students taking the course and their grades of courses taken prior to the course we are going to predict [5].

3.3 Experimental Protocol

3.3.1 Dataset description and preprocessing

We evaluated our proposed methods on two datasets obtained from George Mason University (GMU) and University of Minnesota (UMN), for the following four departments: (i) Computer Science (CS), (ii) Electrical and Computer Engineering (ECE), (iii) Biology (BIOL) and Psychology (PSYC). We will indicate the departments from GMU with the suffix "_A" and from UMN with the suffix "_B". The two universities from two separate states in the United States have different characteristics. For GMU, there are around 33,000 students, the acceptance rate is 69%, the six-year graduation rate is 66.8%, there are about 140 programs that students can select. For UMN, the total enrollment is about 51,000, acceptance rate is 45%, the six-year graduation rate is 75%, there are around 260 programs. Both universities exhibit diversity. In GMU, 44.7% of students are White, 18.5% Asian, 12% Hispanic/Latino, 10% African American. In UMN, 69.1% of the students are White, 11.3% Asian, 5.2% African American, 3.4% Latino.

The data was collected from Fall 2009 to Spring 2016 at GMU and from Fall 2003 to Spring 2014 at UMN. According to the University Catalogs [54] [55], we kept the courses that were required by the degree program and electives within the same major. The statistics of the four majors are shown in Table 3.1.

For UMN that has very flexible degree programs, we also consider courses outside of the department that were taken by at least 50% of the students. We consider those as unstated prerequisites. Moreover, we removed any course that was taken by less than 10% of the students, in order to reduce the size of the universal of courses, i.e., the possible courses that a student might take. We

consider that these courses are not offered on a regular basis and their availability is limited.

For both datasets, we removed any courses whose grades were pass/fail. If a course was taken more than once by a student, only the last grade was kept. We removed the students who took less than half of the prior courses (less than one third of the prior courses for UMN). For course c whose prior-course grade matrix is $\mathbf{G}_c^{pr}$, if the number of rows of $\mathbf{G}_c^{pr}$ is smaller than the number of columns, we remove course c from training and testing dataset. In addition to that, if the number of testing instances of a course is smaller than 5, we also remove it.

To form the test and training dataset, we use the data extracted from last term (i.e., Spring 2016 at GMU and Spring 2014 at UMN) as test dataset and all the data before then as training. The training dataset was split into 80/20, of which 80% was training data, 20% was validation data.

As the flexibility of a degree program can influence the course-specific models' performance, the flexibility associated with each department is computed according to [30]. The major's flexibility is the average course flexibility over all courses belonging to that major, weighted by the number of pairs of students in that offering. We computed the flexibility of a course c as one minus the average Jaccard coefficient of the courses that were taken by the students that took c prior to taking this course. The flexibility of a course will be low if the students have taken very similar prior courses and high otherwise.

To compute the flexibility of a major, assume there are N courses in that major; the prior-course grade matrices for these courses are denoted as $\mathbf{G}_i^{pr}, i = 1 \dots N$, each of which has $S_i, i = 1 \dots N$ students. From matrix $\mathbf{G}_i^{pr}$, we can extract an indicator matrix $\mathbf{I}_i^{pr}$, in which 1 means the corresponding course is taken, 0 means not. $\mathbf{R}_{i,a}$ means the ath row of matrix $\mathbf{I}_i^{pr}$.

$$F_i = 1 - \frac{1}{\binom{S_i}{2}} \sum_{a=1}^{S_i} \sum_{b=a+1}^{S_i} Jaccard(\mathbf{r}_{i,a}, \mathbf{r}_{i,b}) \tag{3.7}$$

$$F = \sum_{i=1}^{N} \frac{S_i}{S} F_i \tag{3.8}$$

where $Jaccard$ is the Jaccard coefficient, S is the total number of students in that major, F_i is the

Table 3.1: Data Statistics and Characteristics for GMU and UMN.

Major	#Students	#Courses	Universal of courses	#Grades	Grades Mn	Grades StD	Flexibility
CS_A	988	18	53	21,880	3.05	0.82	0.283
ECE_A	396	16	69	16,170	3.09	0.77	0.272
BIOL_A	1629	19	42	20,602	3.02	0.84	0.339
PSYC_A	1114	20	60	14,851	3.26	0.74	0.429
CS_B	708	24	39	78,882	3.15	0.71	0.493
ECE_B	551	16	44	86,478	3.12	0.72	0.430
BIOL_B	997	11	31	57,966	3.12	0.74	0.603
PSYC_B	1380	18	37	77,896	3.07	0.82	0.809

#Students is the number of major students.
#Courses is the number of courses for which we predict the grades.
Universal of courses is the total number of prior courses, i.e., the required and elective courses in the corresponding major according to university catalog.
#Grades is the total number of grades in prior-course grade matrices and the grades we predict.
Grades Mn and Grades StD are the mean and standard deviation of grades, respectively.
Flexibility is the flexibility of a major.

flexibility of course i and F is the flexibility of the major.

3.3.2 Evaluation Metrics

To assess the performance of the models, we used three kinds of metrics, namely mean absolute error (MAE), root mean squared error (RMSE) and tick error. MAE and RMSE are computed by pooling together all the grades across all the courses.

MAE and RMSE are averaged errors between the predicted grades and the actual grades. To gain a better insight into the quality of the predictions, we also report the tick error as done in [5,30]. The grading system used in GMU has 11 letter grades (A+, A, A-, B+, B, B-, C+, C, C-, D, F) which correspond to (4, 4, 3.67, 3.33, 3, 2.67, 2.33, 2, 1.67, 1, 0). UMN uses the same grading, with the addition of D+, corresponding to 1.33, and excluding A+. We refer to the difference between two successive letter grades as a tick. The performance of a model is assessed based on how many ticks away the predicted grade is from the actual grade. We first converted the predicted grades into their closest letter grades and then computed the percentages of each of the x ticks [5, 30].

Table 3.2: MAE of different methods (↓ is better).

Method	MAE							
	CS_A	ECE_A	BIOL_A	PSYC_A	CS_B	ECE_B	BIOL_B	PSYC_B
BO	0.7359	0.7285	0.5853	0.5882	0.4697	0.4356	0.4516	0.4648
MF	0.8150	0.8447	0.6169	0.5648	0.4859	0.4309	0.4452	0.4940
CS_{MF}	0.7609	0.7015	0.5579	0.5240	0.4776	0.4433	0.4410	0.4695
CSR_{PC}	0.6805	0.6739	0.5372	0.4933	**0.4520**	0.4346	0.4394	0.4932
CSR_{CF}	0.7183	0.6775	**0.4769**	**0.4743**	0.4670	0.4395	0.4488	0.4588
CSR_{HY}	**0.6693**	**0.6630**	0.5057	0.4859	0.4622	**0.4219**	**0.4328**	**0.4526**

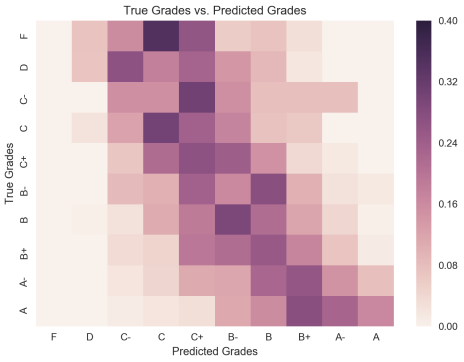
Table 3.3: RMSE of different methods (↓ is better).

Method	RMSE							
	CS_A	ECE_A	BIOL_A	PSYC_A	CS_B	ECE_B	BIOL_B	PSYC_B
BO	0.9622	0.9748	0.7794	0.7829	**0.6534**	0.5359	0.5855	0.6180
MF	1.0879	1.1104	0.8173	0.8035	0.6773	0.5408	0.5922	0.6574
CS_{MF}	1.0126	0.9623	0.8045	0.7372	0.6685	0.5472	0.5763	0.6318
CSR_{PC}	0.9288	0.9699	0.7943	0.7348	0.6613	0.5447	0.5679	0.6351
CSR_{CF}	0.9539	0.9680	**0.7205**	**0.6732**	0.6543	0.5457	0.5825	0.6064
CSR_{HY}	**0.9199**	**0.9542**	0.7679	0.7283	0.6607	**0.5298**	**0.5659**	**0.5946**

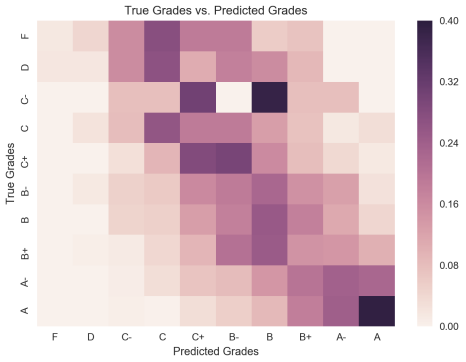
3.4 Results and Discussion

Tables 5.3 and 5.4 show the comparative performance of different methods on four different departments by using metrics MAE and RMSE. Generally, in most cases course-specific models outperform non-course-specific models, which means focusing on a course-specific subset of data can result in better performance. In GMU, for departments with less flexibility such as Computer Science and Electrical Engineering, we observe that the hybrid model has the best performance. Thus incorporating content features into course-specific model further improves its performance; the model with only grades of prior courses performs better than model with only content features. For departments with high flexibility such as Biology and Psychology, the model with only content features shows the best performance, which suggests that if a department has a flexible degree program, content features might be more informative than the grades of prior courses.

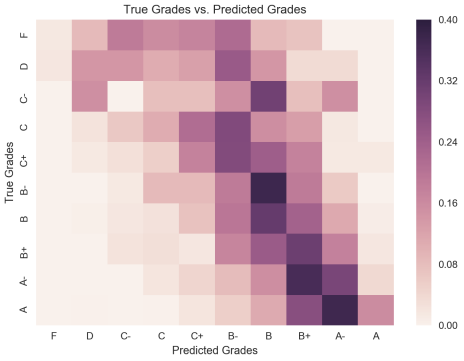
The corresponding departments in UMN are more flexible than GMU. The performance of



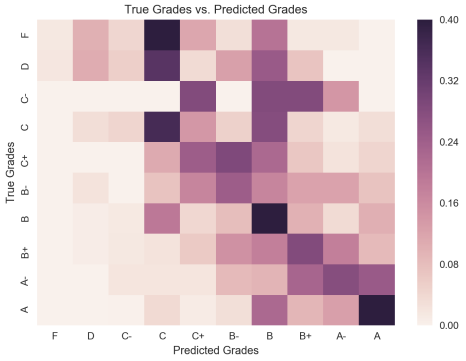
(a) True vs. Predicted Grades for BO



(b) True vs. Predicted Grades for CSR_{PC}

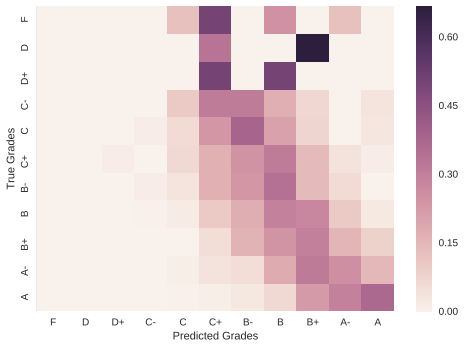


(c) True vs. Predicted Grades for CSR_{CF}

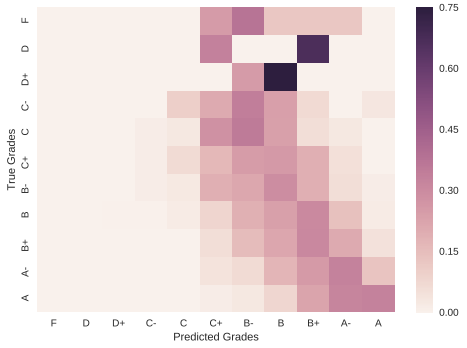


(d) True vs. Predicted Grades for CSR_{HY}

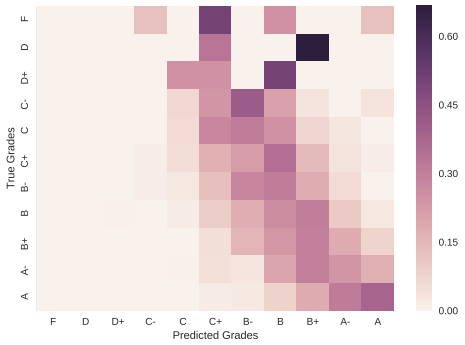
Figure 3.1: True vs. Predicted Grades for BiasOnly and Course-specific Models for GMU.



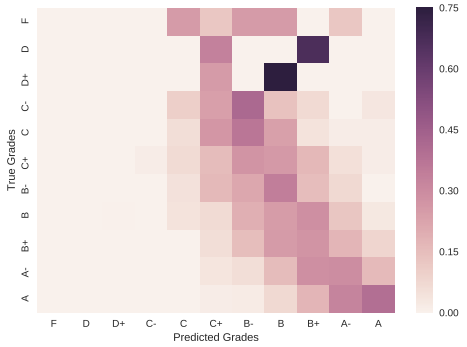
(a) True vs. Predicted Grades for BO



(b) True vs. Predicted Grades for CSR_{PC}



(c) True vs. Predicted Grades for CSR_{CF}



(d) True vs. Predicted Grades for CSR_{HY}

Figure 3.2: True vs. Predicted Grades for BiasOnly and Course-specific Models for UMN.

Table 3.4: Prediction performance of different methods based on Ticks (↑ is better).

#Ticks	Method	CS_A	ECE_A	BIOL_A	PSYC_A	CS_B	ECE_B	BIOL_B	PSYC_B
No tick error	BO	15.02	18.58	19.41	19.75	25.48	27.58	24.90	34.40
	MF	13.04	9.84	19.95	23.89	26.68	28.48	24.90	31.91
	CS_{MF}	15.22	18.58	24.53	23.25	24.76	**29.09**	**30.12**	**34.75**
	CSR_{PC}	19.57	20.77	28.84	34.08	**29.33**	26.06	25.70	23.76
	CSR_{CF}	13.44	16.39	28.03	27.39	25.96	28.48	25.30	31.91
	CSR_{HY}	**19.76**	**22.40**	**30.73**	**35.35**	25.00	28.18	29.32	30.50
One tick error	BO	44.27	44.26	55.26	53.82	65.38	66.36	61.85	65.60
	MF	42.29	39.34	51.75	54.46	63.70	66.67	**65.06**	62.77
	CS_{MF}	43.08	40.44	58.76	61.78	63.94	64.85	**65.06**	**68.44**
	CSR_{PC}	48.22	**55.19**	62.80	61.15	**69.23**	64.85	62.65	57.45
	CSR_{CF}	44.66	51.37	**70.89**	**64.97**	64.42	66.67	63.05	**68.44**
	CSR_{HY}	**49.80**	**55.19**	67.38	61.78	68.03	**66.97**	64.66	66.31
Two ticks error	BO	69.17	66.67	77.63	75.80	86.54	**89.09**	87.15	**88.65**
	MF	64.82	63.38	76.82	77.07	82.69	88.79	86.75	83.69
	CS_{MF}	67.59	72.68	82.21	78.66	85.34	**89.09**	86.75	85.11
	CSR_{PC}	74.31	73.22	81.40	79.62	**87.26**	85.76	**88.35**	83.69
	CSR_{CF}	73.52	**75.96**	**87.87**	**83.44**	86.06	88.79	85.54	87.94
	CSR_{HY}	**75.10**	74.32	82.75	78.66	85.82	88.18	86.35	86.88

CSR_{PC} and CSR_{CF} is very comparable, or even better (for the Psychology Department). Their combination, CSR_{HY}, is the best performing method in terms of MAE and RMSE, even if the content features included in UMN are less informative. An exception is the Computer Science Department, which seams to have very hard-to-predict courses, as it has the highest RMSE. For CS_B, CSR_{PC} is performing the best in terms of MAE, but BiasOnly achieves better RMSE, closely followed by CSR_{CF}, with just 0.0009 difference.

In the two universities, we can see that for the majority of the departments, the hybrid model performs the best. GMU models take more advantage of the rich content features to improve the predicted grades, especially for the most flexible departments.

To gain deeper insights into the types of errors made by different methods, Table 5.5 reports the percentage of grades predicted with no error, with an error of at most one tick and with an error of at most two ticks. Comparing the performance achieved by the methods we notice that the course-specific models have relatively better performance than non-specific approaches. In GMU, in terms of the exact prediction (i.e., no error), the hybrid model has the best performance. For departments

with rigid degree program, such as Computer Science and Electrical Engineering, the hybrid model has better performance than other methods. If minor errors are allowed (i.e., one or two ticks), for flexible departments, model with only content features gives better performance. In UMN, the picture is not that clear, as there is variation in the performance depending on the degree of accuracy and the department. The highest percentage of grades predicted with no error is achieved by course-specific methods(CS_{MF} and CSR_{PC}). The fact that other methods are the best performing in terms of ticks, while CSR_{HY} has the lowest RMSE for most of the cases, indicates that CSR_{HY} does not predict many grades with significant error, in contrast with the other methods.

From the two universities' results, we can see that incorporating content features into the course-specific model can improve the prediction performance. For flexible degree programs, as the prior-course grade matrix is sparse, the model with content features has better predicting accuracy. This is not evident in the results from UMN, as there are not enough content features.

The distribution of true (ground truth) and predicted grades for BiasOnly, CSR_{PC}, CSR_{CF} and CSR_{HY} are also plotted for GMU and UMN in Figures 3.1 and 3.2, respectively. Each row of the figure represents the ratio of the predicted grades. For example, in Figure 3.1b the bottom row represents that a high proportion of A's are predicted as such. We see that BiasOnly tends to smooth the predicted grades i.e., it predicts most of the grades around the average GPA (around B-). However, for high grades most of the predicted grades are around the true grades in course-specific models and for lower grades all the models tend to over predict.

Table 3.5 and 3.6 show the detailed statistics of the courses from the two universities of the departments with the least and most flexible degree program, and the errors (RMSE) made by three course-specific regression models. For GMU these departments are the CS and PSYC, while for UMN are the EE and PSYC. From the two tables, we can see that if the grades in test set have high standard deviation or higher than that of training set, the prediction error is high. The reason might be that the course-specific models used in this work and previous works are linear. In the future, we will explore non-linear course-specific models.

Overall, incorporating content features into the course-specific models can improve the prediction performance. In GMU, for departments with less flexible degree programs, the hybrid model

Table 3.5: Per course statistics and errors for GMU.

Course	#training	#testing	density	Mn Tr	StD Tr	Mn Te	StD Te	CSR_{PC}	CSR_{CF}	CSR_{HY}
CS-2xx	322	76	0.766	2.640	1.249	2.548	1.455	1.179	1.226	1.176
CS-2xx	303	66	0.623	2.915	1.062	2.899	0.941	0.686	0.755	0.735
CS-3xx	138	19	0.748	3.049	0.803	3.158	0.597	0.463	0.417	0.434
CS-3xx	285	62	0.638	2.634	1.155	2.694	1.236	1.037	1.156	1.037
CS-3xx	181	41	0.711	3.063	0.779	3.041	0.617	0.527	0.465	0.539
CS-3xx	42	13	0.802	3.104	1.140	3.360	0.591	0.748	0.668	0.752
CS-3xx	189	35	0.754	2.783	1.032	2.657	1.053	0.876	0.949	0.876
CS-3xx	19	8	0.885	2.719	1.072	2.959	1.368	1.152	1.035	1.253
CS-3xx	156	29	0.768	3.088	0.762	2.897	1.175	1.072	1.045	1.066
CS-4xx	92	8	0.867	2.859	1.103	2.917	1.090	1.006	1.119	1.006
CS-4xx	29	15	0.868	2.426	1.181	2.311	1.341	1.243	0.972	0.830
CS-4xx	35	7	0.378	2.667	0.983	2.713	0.629	0.711	0.609	0.736
CS-4xx	105	36	0.909	3.137	0.810	3.297	0.965	0.951	0.913	0.994
CS-4xx	43	10	0.912	2.923	1.001	2.567	1.383	1.072	1.063	1.042
CS-4xx	46	19	0.896	2.725	1.111	1.983	1.111	1.090	1.081	1.143
CS-4xx	32	8	0.897	3.083	0.866	3.041	1.207	0.964	1.106	0.964
CS-4xx	115	32	0.868	3.018	0.914	3.229	0.659	0.655	0.643	0.655
CS-4xx	26	22	0.868	3.525	0.668	3.333	0.841	0.669	0.870	0.610
PSYC-2xx	195	24	0.608	3.165	0.802	3.639	0.429	0.709	0.604	0.694
PSYC-2xx	204	23	0.635	3.144	0.726	3.435	0.788	0.678	0.746	0.678
PSYC-3xx	247	23	0.670	3.263	0.813	3.580	0.654	0.796	0.656	0.799
PSYC-3xx	223	24	0.724	3.262	0.870	3.390	0.875	0.759	0.578	0.756
PSYC-3xx	44	5	0.825	3.212	0.943	3.600	0.490	0.507	0.829	0.653
PSYC-3xx	112	8	0.613	3.310	0.858	3.292	0.715	0.878	0.726	0.873
PSYC-3xx	86	7	0.558	3.535	0.758	3.620	0.516	0.696	0.467	0.678
PSYC-3xx	258	21	0.586	3.263	0.936	3.778	0.428	0.760	0.801	0.728
PSYC-3xx	69	14	0.718	3.251	0.667	3.357	0.672	0.481	0.475	0.467
PSYC-3xx	227	26	0.687	3.333	0.728	3.270	0.883	0.776	0.729	0.776
PSYC-3xx	94	9	0.723	3.394	0.617	3.630	0.618	0.600	0.521	0.602
PSYC-3xx	52	6	0.714	3.378	0.911	3.280	1.027	0.978	1.033	0.956
PSYC-3xx	216	22	0.731	3.048	0.951	2.803	1.013	0.940	0.642	0.940
PSYC-3xx	66	12	0.710	3.525	0.802	3.168	0.977	1.020	0.865	1.021
PSYC-3xx	121	18	0.715	3.488	0.705	3.371	0.745	0.692	0.627	0.700
PSYC-4xx	182	21	0.672	3.564	0.716	3.540	0.442	0.549	0.346	0.550
PSYC-4xx	48	5	0.789	3.771	0.409	4.000	0.000	0.424	0.253	0.424
PSYC-4xx	105	30	0.661	3.445	0.884	3.778	0.489	0.588	0.627	0.590
PSYC-4xx	34	12	0.798	3.657	0.521	3.112	0.736	0.809	0.893	0.802

The second and third column stand for the number of training and testing instances, respectively
density means the density of the prior course matrix
Tr train, *Te* test, *Mn* mean, *StD* standard deviation

Table 3.6: Per course statistics and errors for UMN.

Course	#training	#testing	density	Mn Tr	StD Tr	Mn Te	StD Te	CSR_{PC}	CSR_{CF}	CSR_{HY}
EExxx	514	22	0.441	2.82	0.73	2.95	0.70	0.603	0.547	0.544
EExxx	511	32	0.450	3.59	0.51	3.44	0.37	0.520	0.664	0.577
EExxx	540	5	0.352	2.88	0.67	2.73	0.25	0.454	0.393	0.419
EExxx	516	28	0.443	2.94	0.68	2.98	0.60	0.562	0.543	0.532
EExxx	520	21	0.405	3.05	0.67	2.84	0.79	0.574	0.589	0.573
EExxx	142	7	0.582	2.83	0.94	2.81	0.24	0.712	0.409	0.599
EExxx	88	31	0.837	3.27	0.63	3.10	0.69	0.559	0.585	0.568
EExxx	146	32	0.631	3.08	0.78	3.04	0.64	0.525	0.449	0.529
EExxx	51	13	0.587	3.86	0.28	3.95	0.18	0.378	0.494	0.383
EExxx	189	20	0.610	2.74	0.81	2.88	0.78	0.548	0.545	0.573
EExxx	225	11	0.576	3.11	0.75	3.12	0.94	0.825	0.835	0.833
EExxx	101	22	0.684	3.06	0.70	2.55	0.56	0.572	0.582	0.579
EExxx	331	29	0.558	3.24	0.63	3.47	0.54	0.445	0.416	0.419
EExxx	239	23	0.556	3.84	0.27	3.91	0.15	0.581	0.414	0.394
EExxx	407	26	0.655	3.65	0.43	3.88	0.45	0.486	0.585	0.485
EExxx	65	8	0.670	3.89	0.37	3.92	0.14	0.149	0.344	0.090
PSYCxxx	1031	18	0.207	3.30	0.59	3.26	0.57	0.452	0.429	0.433
PSYCxxx	464	7	0.259	3.21	0.83	3.14	0.59	1.027	0.998	1.023
PSYCxxx	444	10	0.263	2.90	0.80	3.17	0.43	0.733	0.693	0.784
PSYCxxx	606	17	0.261	3.21	0.77	3.24	0.72	0.490	0.509	0.510
PSYCxxx	557	18	0.254	2.97	0.87	3.48	0.92	0.795	0.794	0.802
PSYCxxx	488	13	0.220	3.03	0.89	3.56	0.48	0.430	0.495	0.438
PSYCxxx	34	12	0.482	2.80	0.90	3.42	0.71	0.873	0.870	0.867
PSYCxxx	399	12	0.259	3.13	0.79	3.39	0.45	0.512	0.389	0.375
PSYCxxx	288	13	0.261	2.97	0.79	2.95	0.76	0.468	0.468	0.485
PSYCxxx	554	7	0.271	3.35	0.67	3.48	0.43	0.471	0.629	0.538
PSYCxxx	743	13	0.162	3.17	0.78	2.87	0.78	0.626	0.812	0.650
PSYCxxx	346	9	0.268	3.30	0.78	2.74	0.91	0.676	0.699	0.705
PSYCxxx	301	10	0.229	3.46	0.59	3.67	0.42	0.907	0.679	0.684
PSYCxxx	366	5	0.276	3.22	0.82	3.07	0.44	0.593	0.660	0.618
PSYCxxx	1045	80	0.354	3.56	0.57	3.70	0.46	0.648	0.601	0.590
PSYCxxx	258	7	0.288	3.90	0.47	4.00	0.00	0.466	0.392	0.343
PSYCxxx	121	5	0.274	3.96	0.16	4.00	0.00	0.194	0.271	0.166
PSYCxxx	290	26	0.320	3.93	0.33	4.00	0.00	0.562	0.341	0.351

The second and third column stand for the number of training and testing instances, respectively.
density means the density of the prior course matrix.
Tr train, *Te* test, *Mn* mean, *StD* standard deviation.

achieves better performance than traditional course-specific models. However, for departments with more flexible degree programs, the grades of prior courses are less informative than content features, therefore, it is more appropriate to include only content features. In UMN, CSR_{HY} achieves the best performance. The existance of some content features can boost the performance of the regression methods when used alone(CSR_{CF}) or in addition to the grades(CSR_{HY}).

3.5 Conclusions

In this paper, we proposed a hybrid model to further improve the performance of the course-specific models. We evaluated the proposed model on datasets from two Universities with different characteristics. The experiments show similar results in the two universities, which suggests the proposed model is generalizable. In conclusion, it is beneficial to incorporate content features into course-specific model, which motivates us to explore other kinds of side information. In the future, we will utilize side information mined from learning management systems.

Chapter 4: Course Specific Markovian Models for Grade Prediction

Several approaches have been developed in the past few years to tackle the problem of next-term grade prediction [4]. In particular, course-specific approaches predicting a student's grade in a course by using the grades on a subset of courses taken prior to the target course [5, 28] have shown promising results. Given the sequential aspect of academic programs; where a chain of courses build fundamental concepts and lead to training (education) of students; these models assume that a subset of related prior courses can provide the necessary knowledge for future courses. Course-specific models are based on regression or matrix factorization. One of the limitations of these course-specific models is that they ignore the temporal dynamics associated with the evolution of a student's knowledge state. The concept of knowledge state is proposed in mathematical psychology literature for assessment of a student's mastery of knowledge. Assessments uncover the particular state of a student and are used for predicting student's future performance and abilities. Latent factor models are useful for modeling students' knowledge state evolution [56].

To model the student learning behavior and predict student's performance we propose the Hidden Markov Model (HMM) and Hidden Semi-Markov Model (HSMM). In these models, students' knowledge states are modeled as hidden states. For HMM, the sojourn time is the number of steps spent in one state before transitioning to another state and is geometrically distributed. However, its performance degrades when the data exhibits long-term temporal dependency [57] as in the case of student knowledge state. For example, a student with strong capability is likely to be a high performing student in the next several semesters, instead of suddenly transitioning to a hidden state indicative of a low performing student. Figure 4.1 shows this property. This figure illustrates the dynamics in students' term GPA across all majors at George Mason University for the first six semesters (excluding Summer terms). We only present full letter grades (i.e., A, B, C, D, F) for this figure. The width of the flow from one semester to another shows the number of grades and illustrates that given a student with a particular GPA in one semester, the GPA in the next semester will

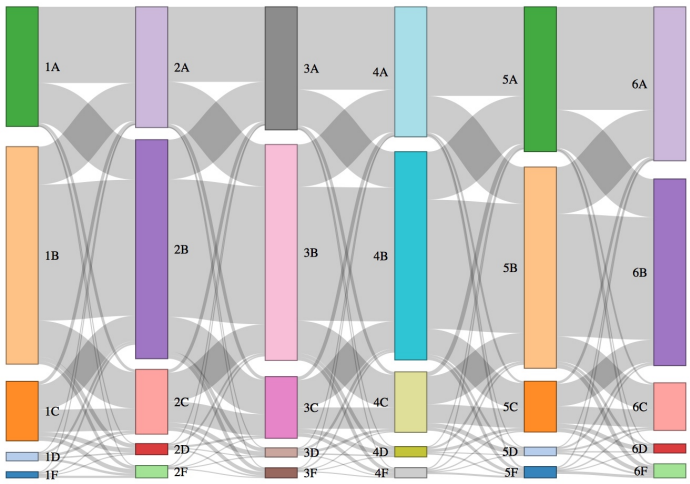
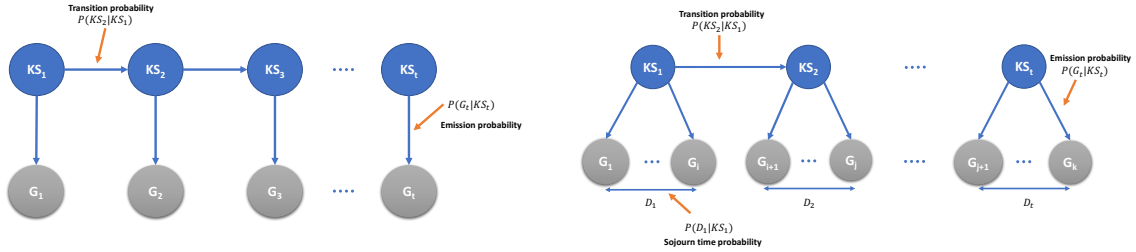


Figure 4.1: Change in Student Term GPA for the first six semesters. The digit of the text label denotes the term and the letter denotes the GPA. E.g., 3B implies term 3 and GPA of B (3.0)

probably remain at the same level or off by one grade point with a high likelihood. If we consider more refined grade points (i.e., the full letter grades plus A+, A-, B+, B-, ..., C-), the statistics of the grade data shows that 24.3% of students have the same GPA from one semester to another, 66.84% and 84.33% of students have their next-term GPA within one and two ticks of their current-term GPA, respectively. Ticks measure the deviations from the true letter grade and is explained in Section 4.4. Thus it is very likely for a student to have similar GPA for the next term, which shows that a student's performance does not change frequently or abruptly.

To capture this long-term dependency property of students' knowledge evolution, we propose HSMM for grade prediction. Compared to HMM, the sojourn time of the knowledge state in HSMM is modeled explicitly. Each hidden state in a HMM emits one observation while in HSMM each hidden state emits a sequence of observations. The number of observations, i.e., the duration d, produced in a hidden state is determined by the sojourn time distribution of that state. Figures 4.2a and 4.2b shows the difference between HMM and HSMM, respectively. In this work, the distribution of sojourn time is assumed to be nonparametric and learned from data.



(a) Graphical Model of the HMM. KS_i represents knowledge state, G_i is the grade in a course.

(b) Graphical Model of the HSMM. KS_i represents knowledge state, G_i is the grade in a course. D_i is the sojourn time of state KS_i.

Figure 4.2: Graphcial model of HMM vs. HSMM

4.1 Notations

Table 4.1: Notations

Symbol	Description
O	Observation
KS	Knowledge State
π	Initial state distribution
D	The maximum number of duration of the hidden states
$g_{s,c}$	the true grade of student s in course c
$\hat{g}_{s,c}$	the predicted grade of student s in course c
a_{ij}	The probability of transition from state i to state j
$b_j(G_t)$	The probability of observing G_t at state j
$d_j(u)$	The probability of staying at state j for u steps

4.2 Problem Formulation and Notations

Assume that we have records of n students and m courses; comprising a $n \times m$ sparse grade matrix $\mathbf{G}$. Entry $G_{s,c}$ in $\mathbf{G}$ represents the grade of student s in course c. In addition we have the time stamp information for each grade $G_{s,c}$. Besides the grade matrix $\mathbf{G}$, we have information associated with the student (e.g., academic level, previous GPA and major) and course offering (e.g., discipline,

course level and difficulty) that can be combined to extract features. We denote the feature vector as **x** of p dimensions. As a convention, bold uppercase letters are used to represent matrices (e.g., **X**) and bold lowercase letters represents vectors (e.g., **x**).

4.3 Methods

4.3.1 Hidden Markov Model (HMM)

Model Description

HMM seeks to capture the dynamic evolution of student's knowledge state. Student's knowledge state is modeled as the latent (hidden) states in HMM. The grades of a student are modeled as the observations. Compared to existing discriminative models, one of the key advantages of the HMM approach is that it introduces stochasticity/uncertainty. For example, a student with high capability has the chance to get a low grade by slipping [20], which is hard to model using discriminative models.

In HMM, the evolution of student's knowledge state is modeled as a Markov chain and has the assumption that the next state only depends on current state. The transition distribution of the model determines the evolution of students' knowledge state, as shown in Equation 4.1.

$$a_{ij} = P(\mathrm{KS}_{t+1} = j | \mathrm{KS}_t = i) \tag{4.1}$$

The emission distribution determines a student's performance, given his knowledge state, given by Equation 4.2.

$$b_j(\mathrm{G}_t) = P(\mathrm{G}_t | \mathrm{KS}_t = j) \tag{4.2}$$

where G_t is the student's grade at time t.

A student's knowledge state cannot be observed; only the grades are observable. The space of the knowledge states and the observations are discrete.

To use HMM for modeling student's knowledge state evolution and predict performance in next course, two related questions need to be answered:

- Given an observation sequence and a model, what is the likelihood of the observation sequence? This question can be solved by using forward algorithm [33].
- Given a set of sequences, in our case they are the sequence of students' grades, how do we infer the parameter set of the model? This can be done by using the classical EM algorithm [33].

Grade Prediction

To predict the grade $\hat{g}_{s,c}$ for student s in a future course c, we first extract the grades of student s in a series of courses $c_1, c_2, ..., c_T$ taken prior to course c and form them as a sequence $\vec{G}_s = g_{s,c_1}, g_{s,c_2}, ..., g_{s,c_T}$. Assume that there are N possible grades student s could get in course c, in our case, the possible grade x is in (4, 4, 3.67, 3.33, 3, 2.67, 2.33, 2, 1.67, 1, 0). Then we have the following:

$$\begin{aligned} \hat{g}_{s,c} &= \max_x P(x|g_{s,c_1}, g_{s,c_2}, ..., g_{s,c_T}) \\ &= \max_x \frac{P(g_{s,c_1}, g_{s,c_2}, ..., g_{s,c_T}, x)}{P(g_{s,c_1}, g_{s,c_2}, ..., g_{s,c_T})} \\ &\propto \max_x P(g_{s,c_1}, g_{s,c_2}, ..., g_{s,c_T}, x) \end{aligned} \tag{4.3}$$

The grade $\hat{g}_{s,c}$ is predicted using maximum likelihood.

4.3.2 Hidden Semi-Markov Model (HSMM)

Model description

The HMM model proposed above assumes that a single knowledge state emits grade distributions for one course only. Further, the number of time steps spent in a given state (i.e., sojourn time) in a

HMM model has geometric distribution as shown by Equation 4.4 [58].

$$\begin{aligned} d_i(u) &= P(S_{t+u+1} \neq i, S_{t+u} = i, S_{t+u-1} = i, ..., \\ & S_{t+2} = i | S_{t+1} = i, S_t \neq i) \\ &= a_{ii}^{u-1}(1 - a_{ii}) \end{aligned} \tag{4.4}$$

where $d_i(u)$ is the probability of staying at state i for u steps.

However, a student's knowledge state has long-term temporal dependency. It is demonstrated that a student with strong academic capabality is unlikely to become low performing in a short time.

To better model student's knowledge state evolution, we propose Hidden Semi-Markov Model (HSMM) shown in Figure 4.2b. For HSMM, the underlying process is assumed to be a semi-Markov chain. Each state can emit variable number of observations. In other words, each knowledge state is responsible for performance in multiple courses. The sojourn time of HSMM is explicitly modeled and different hidden states have different sojourn time distribution. For modeling student's knowledge state evolution, the sojourn time distribution for a given knowledge state j is defined as following.

$$d_j(u) = P(\text{KS}_{t+u+1} \neq j, \text{KS}_{t+u-v} = j, v = 0, ..., u-2 | \text{KS}_{t+1} = j, \text{KS}_t \neq j) \tag{4.5}$$

which is assumed to be nonparametric in this work (i.e. categorical distribution). The state transition distribution of the semi-Markov chain determines the evolution of knowledge state; and is show in Equation 4.6.

$$a_{ij} = P(\text{KS}_{t+1} = j | \text{KS}_{t+1} \neq i, \text{KS}_t = i) \tag{4.6}$$

where $j \neq i$, $\sum_{j \neq i} a_{ij} = 1$ and $a_{ii} = 0$.

The emission distribution of HSMM determines a student's performance, given their knowledge state. Similar to HMM, for student's knowledge state modeling and grade prediction we need to compute the likelihood of a sequence and infer the parameters of HSMM which can be done by

using forward and EM algorithms, respectively. The prediction of a student's grade in a future course by using HSMM is the same as HMM shown in Equation 4.3.

4.3.3 Baseline Methods

Bias Only (BO) The Bias Only method only takes into consideration student's, course's and global bias. The predicted grade is estimated using 4.7.

$$\hat{g}_{s,c} = b_0 + b_s + b_c \tag{4.7}$$

where b_0, b_s and b_c are the global bias, student bias and course bias respectively.

Matrix Factorization (MF) The use of MF for grade prediction is based on the assumption that the students' and courses' knowledge space can be jointly represented in low-dimensional latent feature space [5]. The grade is estimated as:

$$\hat{g}_{s,c} = b_0 + b_s + b_c + \boldsymbol{p}_s^T \boldsymbol{q}_c \tag{4.8}$$

where $\boldsymbol{p}_s$, $\boldsymbol{q}_c$ are the latent vectors representing student s and course c, respectively. We also applied course-specific matrix factorization (CS_{MF}) for grade prediction, which utilizes a course-specific subset of data to estimate a matrix factorization model [28].

Course-Specific Regression with Prior Courses (CSR_{PC}) CSR_{PC} predicts the grade of a student s in a future course c as a sparse linear combination of grades in the courses taken prior to course c [28].

Course-Specific Regression with Content Features CSR_{CF} predicts the performance of a student in a course using content features such as academic level, difficulty level and instructor information [59].

Course-Specific Hybrid Model (CSR_{HY}) .

This model is obtained by combining the content feature vector and prior course vector [59].

4.4 Experiments

4.4.1 Dataset description and preprocessing

We evaluate the proposed methods on a dataset from George Mason University, the largest public university in Virginia and enrolled about 36000 students in Fall 2017. We extracted student and course related data from the largest five undergraduate majors in terms of student enrollment. These included: (I) Computer Science (CS), (II) Electrical and Computer Engineering (ECE), (III) Biology (BIOL), (IV) Psychology (PSYC) and (V) Civil Engineering (CEIE).

We used data from the period of Fall 2009 to Spring 2016. Using the University catalog [54] we selected student records for courses that are required by the major program and electives offered by the department offering the major. We also removed courses that did not result in a grade score (in between A-F) but were only pass/fail courses. If a course was taken more than once by a student, only the last grade was kept. For a course, if the number of students who had taken the course was smaller than the number of the prior courses of this course we removed this course from training and test sets. If the number of test instances of a course was smaller than 5, we removed it.

To simulate the real-world scenario of predicting the next-term grades for students we use the data extracted from the latest term as testing data and all the data from terms prior to the latest term as the training set. The training data was split into 80/20, of which 80% was training data and 20% was validation data for choosing the hyperparameters associated with the model. After selection of hyperparameters, the model was retrained on the entire training set before final evaluation on the last term (test set).

4.4.2 Evaluation Metrics

The performance of the methods were assessed by three different evaluation metrics: (i) mean absolute error (MAE), (ii) root mean squared error (RMSE) and (iii) tick error. MAE and RMSE are computed by pooling together all the grades across all the courses.

To gain deeper insights regarding the performance of the methods for course selection and degree planning, we report an application-specific metric called tick errors [5]. Tick error measures

Table 4.2: Comparative Performance of different models using MAE. (↓ is better)

Method	MAE				
	CS	ECE	BIOL	PSYC	CEIE
BO	0.7257	0.6902	0.5411	0.5951	0.5863
MF	0.7184	0.6790	0.5420	0.6099	0.5796
CS_{MF}	0.7151	0.6666	0.5365	0.5673	0.5733
CSR_{PC}	0.6805	0.6739	0.5372	0.4933	0.601
CSR_{CF}	0.7183	0.6775	0.4769	0.4743	0.6091
CSR_{HY}	0.6693	0.6630	0.5057	0.4859	0.5839
CS_{HMM}	0.601	0.4532	0.4634	0.3362	**0.3632**
CS_{HSMM}	**0.555**	**0.3782**	**0.4231**	**0.3023**	0.3676

the deviation of the predicted grades from the true grades. The performance of a model is assessed based on how many ticks away the predicted grades are from the actual grades. The grading system has 11 letter grades (A+, A, A-, B+, B, B-, C+, C, C-, D, F) which correspond to (4, 4, 3.67, 3.33, 3, 2.67, 2.33, 2, 1.67, 1, 0). To compute the tick error for a predicted grade, the real value prediction outputs are first converted to the closest letter grades.

4.5 Results and Discussion

4.5.1 Comparative Performance

Table 5.3 shows the average MAE for the different methods across the five majors on the test set. The results show that the CS_{HSMM} model achieves the best performance on all the majors. The CS_{HMM} outperforms previously developed Course-Specific regression and factorization models. The proposed course-specific Markovian models are able to take into account the temporal dynamics associated with the evolution of student's knowledge states in comparison to prior course-specific approaches. The CS_{HSMM} model outperforms CS_{HMM} on almost all the majors and has similar performance on CEIE. The students' knowledge state which is modeled by CS_{HMM} and CS_{HSMM} as hidden states tends to stay in the same state for some time instead of changing constantly. Rather than a geometric distribution, the ideal duration distribution should have lower probabilities on

Table 4.3: Comparative Performance of Different Models using Tick Error (↑ is better)

#Ticks	Method	CS	ECE	BIOL	PSYC	CEIE
Zero tick	BO	6.71	10.38	15.36	27.07	20.08
	MF	7.11	13.11	15.90	28.34	21.26
	CS_{MF}	6.72	12.57	16.17	28.98	21.26
	CSR_{PC}	19.57	20.77	28.84	34.08	27.17
	CSR_{CF}	13.44	16.39	28.03	27.39	29.13
	CSR_{HY}	19.76	22.40	30.73	35.35	26.38
	CS_{HMM}	**37.78**	45.36	43.13	**54.24**	**50.39**
	CS_{HSMM}	37.18	**46.99**	**46.49**	52.2	49.78
One tick	BO	29.84	33.33	28.84	45.54	35.83
	MF	29.84	31.15	29.65	43.95	34.25
	CS_{MF}	30.24	33.87	29.38	45.86	34.65
	CSR_{PC}	48.22	55.19	62.80	61.15	52.76
	CSR_{CF}	44.66	51.37	**70.89**	64.97	52.76
	CSR_{HY}	49.80	55.19	67.38	61.78	53.15
	CS_{HMM}	54.08	68.85	66.58	79.32	**72.44**
	CS_{HSMM}	**57.85**	**78.14**	68.65	**79.66**	72.29
Two ticks	BO	60.67	58.47	62.80	69.75	58.66
	MF	59.88	57.92	61.46	68.15	57.48
	CS_{MF}	61.26	59.02	61.19	71.97	57.48
	CSR_{PC}	74.31	73.22	81.40	79.62	69.69
	CSR_{CF}	73.52	75.96	**87.87**	83.44	66.14
	CSR_{HY}	**75.10**	74.32	82.75	78.66	69.29
	CS_{HMM}	70.97	82.51	83.29	85.76	86.22
	CS_{HSMM}	72.37	**87.98**	84.32	**88.81**	**87.01**

longer or shorter durations but higher probabilities on medium durations. By modeling the transition of hidden states as semi-Markov model rather than Markov model, CS_{HSMM} achieves better performance than CS_{HMM}. The exception on CEIE is because of the flexiblility in the particular degree program (i.e., there are many electives for students to choose).

To have better insights into what kind of errors different methods make, we evaluate the approaches using tick error metrics. Table 5.4 presents the results of the best performed traditional course-specific methods (i.e., CSR_{PC} and CSR_{HY}) and the proposed Markovian methods with respect to tick errors. For exact prediction (i.e., 0 tick error) and one tick error, the CS_{HMM} and CS_{HSMM} have the best performance. For two tick errors, the CS_{HMM} and CS_{HSMM} win for most

Table 4.4: Predictive Power at Identifying At-Risk Students (↑ is better)

Method	CS		ECE		BIOL		PSYC		CEIE	
	Acc	F-1	Acc	F-1	Acc	F-1	Acc	F-1	Acc	F-1
CSR_{PC}	0.8202	0.5561	0.7869	0.4507	0.8437	0.5397	0.9172	0	0.7143	0.5217
CSR_{HY}	0.8063	0.5333	0.7923	0.4412	0.8491	0.5625	0.9204	0	0.7532	0.6122
CS_{HMM}	0.8231	0.6276	0.8634	0.7126	**0.9027**	**0.7831**	**0.9492**	0.4828	**0.9004**	**0.6462**
CS_{HSMM}	**0.8549**	**0.7092**	**0.8962**	**0.7711**	0.8784	0.7594	0.9458	**0.5294**	**0.9004**	**0.6462**

The percentage of at-risk students for each major is CS (24.40%), ECE (19.14%), BIOL (18.79%), PSYC (9.80%), CEIE (12.97%).

of the majors, while traditional course-specific model CSR_{HY} show better performance in CS majors. The traditional course-specific models are poorer than CS_{HMM} and CS_{HSMM}, as they ignores students' knowledge evolution dynamics. The reason that CSR_{CF} and CSR_{HY} show better performance in some cases is that they incorporate content features which are informative for student's performance prediction and are not included within the Markovian approaches proposed here.

4.5.2 Case Study: At-Risk Students

An important application of grade prediction is to develop an early-warning system that is able to identify students at-risk of failing the courses that they plan to enroll in. We define at-risk students as those whose grade for a course is below 2.0. To assess the capability of the methods on catching at-risk students we treat the prediction as a classification problem. The experimental procedures are similar to grade prediction as discussed in Section 4.4.1 except that the predicted grade over 2.0 are treated as pass and below 2.0 as fail. We compare the best performed traditional course-specific methods CSR_{PC} and CSR_{HY} with models proposed here. The evaluation metrics are chosen as accuracy and F-1 score. Given the imbalanced nature of the dataset, F-1 score is a suitable classification metric. From Table 5.5, we see that the proposed CS_{HMM} and CS_{HSMM} outperform all the baseline methods. In most cases, the CS_{HSMM} outperforms the CS_{HMM} models. For the Psychology major (has the lowest proportion of at-risk students as shown by numbers in the table notes), some of the existing methods are not able to identify any of the at-risk students and their F-1 score is zero.

4.6 Conclusions

In this paper, we propose Course-Specific Hidden Markov Model and Hidden Semi-Markov Model for student's next-term grade prediction. The proposed Markovian models are able to capture the temporal dynamic characteristics of students' knowledge state evolution. The limitation of HMM is that its hidden state duration is inherently geometrically distributed. To better model student's knowledge state evolution, we use Hidden Semi-Markov Model for grade prediction to model the distribution of state duration explicitly.

We conducted extensive experiments and compared the proposed Markovian models with other state-of-the-art grade prediction algorithms. The experimental results showed that the proposed models achieved better grade prediction performance than the baselines. One important application of grade prediction is early-warning systems. We evaluated the performance of the proposed methods for identifying at-risk students. For this task, our proposed methods achieved the best performance in comparison to other state-of-the-art methods.

Chapter 5: Grade Prediction with Uncertainty Estimation Using Bayesian Deep Learning

From the perspective of educational psychology, learning is affected by both external and internal factors such as motivation, study habits, attention and instructor pedagogy [10, 60], which bring about challenges for grade prediction. These challenges are further exacerbated by the fact that learning is a reflection of human cognition which is a complex process [42]. Existing course-specific models are linear shallow learners, e.g. linear regression or low-rank matrix factorization. These shallow learners may not be capable of capturing complex interactions underlying student's learning. To better model students' learning process, we propose to use deep learning models. Another drawback of traditional grade prediction methods is that the predicted grade is a point estimation. To make informed decision based on the predicted results, we need to know if the prediction system is confident or not. If the system is confident enough about the predictions, we can rely on them and take corresponding actions. However, if the prediction is not reliable, human advisors should decide what to do. Compared to traditional deep learning models, Bayesian deep learning models can provide principled uncertainty estimation, which is useful for deciding if a system is confident or not.

Specifically, we propose two types of Bayesian deep learning models, (I) Multilayer Perceptron (MLP) [61], (II) Long Short Term Memory (LSTM) networks [62]. MLP consists of hierarchical hidden layers that maps the input vector to an output target. The input vector is treated as static and hence the temporal dynamics of the input data are ignored by MLP. To capture student's knowledge evolution, we also propose a LSTM model. Theoretically, RNNs are able to model arbitrarily long sequential data. However, in practice, because of the vanishing gradient problem, vanilla RNNs fail to capture long-term dependencies. For grade prediction, a course taken several semesters ago might still have influence on the student's performance in a future course. To model such long-term dependencies, we choose to use LSTM model.

To trust the predictions from a model, we need to know if the system is confident about its predictions or not. We provide empirical results about model uncertainty and investigate case studies towards developing a reliable educational early warning system. We also propose a method to explain the models' predictions, which identifies a list of influential prior courses that lead to a student's failure in the target course.

The main contributions of this work can be summarized as follows:

- We propose two types of course-specific Bayesian deep learning models for grade prediction, namely, course-specific MLP and LSTM. Compared to existing methods, the proposed models have better modeling capability and prediction accuracy.
- The proposed models can provide prediction uncertainty which is essential for decision making. Based on uncertainty estimation, we show how uncertainty can help build a reliable educational early warning system.
- In addition to uncertainty estimation, we propose a method to explain the prediction results, which can identify influential courses that results in a student's failure of a course.
- We propose a method to evaluate the models' capability of catching at-risk students. The evaluation results show that the proposed methods outperform several baseline methods for this task.

5.1 Methods

5.1.1 Model Learning Framework

Given records of n students and m courses, we extract the grades to form a sparse grade matrix $\mathbf{G} \in \mathbf{R}^{n \times m}$. In addition, we have the information associated with the semester (time) when the particular grade was obtained. Further, the data includes student-related features (e.g., academic level, previous GPAs, major, etc.) and course-related features (e.g., course level, discipline, credit hours, etc.). These content features are combined to form a feature vector associated with a student-course pair.

Given a student's grades in the courses taken before the target course (referred to as prior courses), the objective of the next-term grade prediction problem is to predict the grade that the student will achieve in a course to be taken in the next semester (term). To predict grade in a course-wise manner, we adopt course-specific framework [28]. Under this framework, different models are learnt for different courses. To predict a student's grades in next courses, his/her grades from prior courses are fed into corresponding models.

5.1.2 Multilayer Perceptron

Traditional grade prediction models are linear models, such as linear regression. Compared to linear models, the key advantage of multilayer perceptron comes from its hierarchical hidden layers that capture complex interactions and non-linearities. The theoretical foundation is given by the Kolmogorov-Arnold representation theorem [63, 64]; every multivariate continuous function can be represented as a superposition of one-dimensional continuous functions.

Given an input vector $\boldsymbol{x}$, the task of the multilayer perceptron algorithm is to map $\boldsymbol{x}$ to output $\boldsymbol{y}$, which has the following form

$$\boldsymbol{y} = F(\boldsymbol{x}) \tag{5.1}$$

To estimate a student's grade in course c by using the course-specifc MLP model F^c, we have

$$\hat{y}^c = F^c(\mathbf{s}) \tag{5.2}$$

where $\mathbf{s} \in \mathbb{R}^m$ is the vector of the student's grades in the prior courses.

5.1.3 Long Short Term Memory

To capture the sequential characteristics of students' grades in prior courses, we model the learning behavior and performance using recurrent neural networks with long short term memory [62] (LSTM). The standard RNN model has the vanishing gradient problem and is unable to capture long-range dependencies. In our case, an course taken several semesters before, such as a prerequisite, plays an important role in determining a student's performance in a target course. To solve

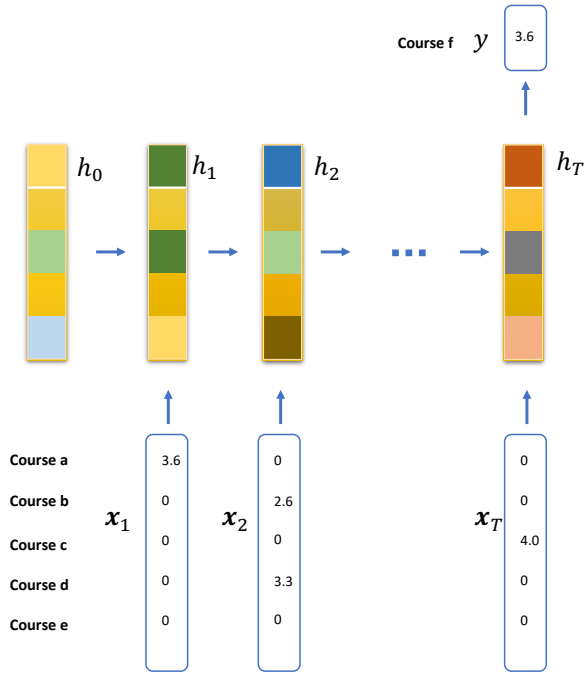


Figure 5.1: In this example, we want to predict a student's grade in target course f by using grades of the courses taken prior to course f include a, b, c, d, e. $\mathbf{x_t}$ represents the grades of courses in term t. y represents the predicted grade. The student took courses (a), (b, d), ..., (c) in semester 1, 2, ..., T and obtained (3.6), (2.6, 3.3), ..., (4.0) in this example, respectively.

the long-term dependency problem, LSTM is proposed for sequential data. The hidden states of LSTM capture the student's knowledge states, which models a student's knowledge evolution. The hidden states are updated as the student enrolls for courses and obtain grades in them. Figure 5.1 shows the LSTM approach for modeling student's learning process. At the beginning, a student has some prior knowledge before taking any courses; the student's knowledge states evolve as he/she take courses, as indicated by different colors at each time step in Figure 5.1. A student's knowledge states influences his/her performance in a course .The last hidden state $\mathbf{h}_T$ is used to predict his/her grade in a target course within the course-specific framework.

LSTM is a gated recurrent neural network, which consists of forget gate and input gate. The forget gate decides which part of the information to forget from the cell state. This is useful when the same knowledge can be obtained by taking two different courses. A student's knowledge state corresponding to knowledge acquired by taking the first course can be discarded while renewed by

using the second one. When student takes a new course, his/her knowledge state is updated. In LSTM, this is done by the information layer and input gate; input gate decides which new information should be added into the cell state. The output from LSTM is hidden state which represent student's current knowledge state.

To estimate a student's grade in the target course by using LSTM model, we first extract the student's grades in the prior courses with timestamp i.e., in which terms the prior courses are taken. The grades in term t are represented as multiple-hot encoded vector $\mathbf{x}_t$ — as more than one course can be taken together in one semester — where the entries of $\mathbf{x}_t$ corresponds to the grades of courses taken in semester t; 0 represents the corresponding courses are not taken. If the grade obtained is 0 (F), we use a small number (0.1) to represent it to differentiate it from courses that are not taken. We input the sequence of the encoded vectors $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T$ to the model and the hidden state from the last step $\mathbf{h}_T$ is fed into a fully connected layer, the output of which is the predicted grade:

$$y = \mathbf{w}_{yh} \cdot \mathbf{h}_T + b_y \tag{5.3}$$

where $\mathbf{h}_T$ is the last hidden state, $\mathbf{w}_{yh}$ is the parameters of the fully connected layer and b_y is the bias term.

Table 5.1: Dataset Statistics

Major	Fall 2016			Spring 2017		
	#S	#C	#G	#S	#C	#G
CS	2,664	18	22,246	3,728	19	33,039
ECE	1,160	16	16,415	1,421	15	23,459
BIOL	2,736	19	20,984	6,002	20	42,895
PSYC	2,980	20	14,966	4,628	20	23,560
CEIE	1,525	18	23,954	1,873	17	28,198
Overall	11065	91	98,565	17652	91	151,151

#S number of students, #C number of courses, #G number of grades

5.1.4 Uncertainty Estimation

Given input data $\mathbf{x}$, the output of an Bayesian deep learning model $f(\cdot)$ is mean $\hat{y}$ and standard deviation σ, where σ is treated as uncertainty; the lower the standard deviation, the higher the predictive confidence. Bayesian models such as Gaussian Process provide principled uncertainty estimation, however, they are computationally prohibitive and hard to scale to large-scale datasets. Yarin et al., showed that dropout can be interpreted as a Bayesian approximation and Monte Carlo (MC) dropout is proposed to obtain prediction uncertainty [11]. Dropout is first proposed as a method for preventing overfitting in neural networks. The basic idea of MC dropout is that for each input, we repeat the prediction for T iterations to get T different outputs, at each iteration neurons are randomly set to zero with some dropout probability. In the next section, we describe how to obtain uncertainty by using Monte Carlo dropout.

Given an deep learning model trained with dropout probability p, we sample T sets of model parameters $W_1, W_2, ..., W_T$ with different dropout masks to have different model realizations $f^{W_1}, f^{W_2}, ..., f^{W_T}$. For an input $\mathbf{x}_i$, the outputs from T model realizations are

$$\hat{y}_i^t = f^{W_t}(\mathbf{x}_i) \tag{5.4}$$

The prediction mean $\overline{y}$ is estimated as

$$\overline{y} \approx \frac{1}{T}\sum_{t=1}^{T}\hat{y}_i^t \tag{5.5}$$

The prediction variance is estimated as

$$\sigma_i^2 = \tau^{-1} + \frac{1}{T}\sum_{t=1}^{T}(\hat{y}_i^t)^2 - \overline{y}^2 \tag{5.6}$$

which equals sample variance plus model uncertainty τ^{-1}, where τ is a hyperparameter which needs to be tuned for different datasets [11].

Given prediction mean $\overline{y}$ and variance σ_i^2, an α-level prediction interval is calculated as

$$[\overline{y} - z_{a/2}\sigma,\ \overline{y} + z_{a/2}\sigma] \tag{5.7}$$

where $z_{a/2}$ is the upper $(1-C)/2$ critical value for standard normal distribution. For example, 95% prediction interval can be calculated as $[\overline{y} - 1.96\sigma,\ \overline{y} + 1.96\sigma]$.

5.1.5 Interpretability

When a model is used for decision making, it is necessary for practitioners to have confidence in the predictions in order to act upon them [65–67]. When an instructor is notified of an at-risk student, they need to know not only the predicted grades but also the reasons associated with the corresponding predictions (e.g., which prior courses lead to a student's failure in the target course). Towards this end, we develop an approach to explain the predictions made by the proposed model. The course-specific model assumes that the knowledge needed for a course is accumulated when taking the prior courses. As such, one of the factors associated with student's performance is his grade/performance in the prior courses. We compute the influence of a prior course in the following way. Given a trained model M and a student s, the grade predicted by the model for this student is denoted as $\hat{y}_s$. Let p be a prior course and $\hat{y}(\neg p)_s$ be the predicted grade if the corresponding grade of course p is set to full grade, namely, 4.0 in the input to the model. For student s, the influence of course c — denoted by $I_c(s,p)$ — is computed as

$$I_c(s,p) = \hat{y}(\neg p)_s - \hat{y}_s \tag{5.8}$$

The intuition behind this approach is that if a student could have obtained higher grades in a prior course, he/she is likely to have better performance in the target course. Based on this information a student could be advised to prepare or review the material in these influential courses so as to be successful in the target course.

This can also be used to improve the curriculum structure. By considering students collectively, if there exists a prior course that consistently has a high influence for a target course across several

students, then this prior course material needs to be a prerequisite or reviewed in class (if not already present). To compute the influence of a prior course on a target course, we observe that different students have a different grade in a prior course. Instead of setting the grade of a prior course to 4.0, we increase its grade by a fixed value of 1.0 The following equations describe how to compute the influence of a prior course on a target course.

$$I_c^*(s,p) = y^*(p_{+1.0}) - \hat{y}_s \tag{5.9}$$

$$I_c(p) = \sum_{s \in S} I_c^*(s,p) \tag{5.10}$$

where p is the prior course, S is all the students that have taken target course c, $y^*(p_{+1.0})$ is the predicted grade if grade of prior course p is increased by 1.

5.2 Experimental Protocols

5.2.1 Dataset Description

The methods are evaluated on a dataset from University X. We choose the largest five undergraduate majors including: (i) Computer Science (CS), (ii) Electrical Engineering (ECE), (iii) Biology (BIOL), (iv) Psychology (PSYC) and (v) Civil Engineering (CEIE). To build a course-specific model for a target course, we choose the prior courses according to the University Catalog from Fall 2009 to Spring 2017.

The evaluation simulates the real-world scenario of predicting the next-term grades for students. Specifically, the models are trained on the data up to term $T-1$ and tested on term T. The last two terms are chosen as testing terms, i.e., Fall 2016 and Spring 2017. As an example, to evaluate the performance of predictions on term Fall 2016, the model is trained on data from Fall 2009 to Fall 2015; and data from Spring 2016 is used for selecting hyperparameters. Dataset statistics are in Table 5.1.

Table 5.2: Tick error example

True Grade	Predicted Grade	Tick Error
B	B	= 0
B	B-, B, B+	≤ 1
B	C+, B-, B, B+, A-	≤ 2

5.2.2 Model Training

The deep learning models are trained by using the Adam [68] optimizer. For the hyperparameters, we use the grid search to choose the best combination on the validation dataset as described above. Every 50 iterations, we take a snapshot of the model and the model that performs the best on the validation dataset is selected for final evaluation on the test set. The hyper-parameters for MLP include the number of layers (ranging from 2 to 10) and the number of neurons in each of the hidden layers (ranging from 2 - 50). For the stacked-LSTM, the parameters include the number of hidden dimensions (ranging from 10 to 100) and the number of stacked layers (ranging from 1 to 5). The activation function is Rectified Linear Unit (ReLU) and the learning rate was set to 0.001. The configuration parameters for Adam are set to default values (β_1 is 0.9, β_2 is 0.999 and ϵ is 10e-8).

5.2.3 Evaluation Metrics

We employ different evaluation metrics including the Mean Absolute Error (MAE) and Percentage of Tick Accuracy (PTA) [5]. In the grading system, there are 11 letter grades (A+, A, A-, B+, B, B-, C+, C, C-, D, F) which correspond to (4, 4, 3.67, 3.33, 3, 2.67, 2.33, 2, 1.67, 1, 0). A tick is defined as the difference between two consecutive letter grades. The performance of a model is assessed based on how many ticks away the predicted grade is from the true grade. For example, a true grade of B vs. prediction of B is zero tick; true grade of B vs prediction of B- is one tick and true grade of B vs a prediction of C+ is two ticks. Table 5.2 shows an example. To assess the performance of the models by using PTA, we first convert the predicted numerical grades to the closest letter grades and then compute the percentages of each of the x ticks.

5.2.4 Comparative Methods

We compare the proposed models with different approaches including matrix factorization and the traditional course-specific models.

Bias Only (BO)

The Bias Only method only takes into account a student's bias, course's bias and global bias [59]. The predicted grade $\hat{g}_{s,c}$ by using this model is estimated as

$$\hat{g}_{s,c} = b_0 + b_s + b_c \tag{5.11}$$

where b_0, b_s and b_c are the global bias, student bias and course bias, respectively.

Matrix Factorization (MF)

To use matrix factorization for a student's grade prediction, we assume that students and courses can be jointly represented in low-dimensional latent space [59]. The grade of student s in a future course c can be predicted as

$$\hat{g}_{s,c} = b_0 + bs_s + bc_c + \mathbf{p_s^T q_c} \tag{5.12}$$

where b_0, b_s and b_c are the global bias, student bias and course bias, respectively; $\mathbf{p_s}$ and $\mathbf{q_c}$ are latent vectors corresponding to student s and course c.

Course-Specific Regression with Prior Courses

The course-specific regression with prior courses (CSR_{PC}) [28] predicts the grade of a student s in a course c as a linear combination of the grades in prior courses. The predicted grade $\hat{g}_{s,c}$ is

$$\hat{g}_{s,c} = w_{c0} + \mathbf{x_{s,c}^T w_c^{pr}} \tag{5.13}$$

where $\mathbf{x_{s,c}} \in \mathbf{R^{m_c}}$ is a feature vector encoding the grades of prior courses, w_{c0} is the bias term and $\mathbf{w_c^{pr}} \in \mathbf{R^{m_c}}$ is the weight vector to be learned and m_c is the number of prior courses.

Course-Specific Regression with Content Features

The Course-Specific Regression with Content Features (CSR_{CF}) model [59] predicts a student's grade in a course using content features related to the student (e.g., academic level, previous GPAs, major, etc) and the course (e.g., course, discipline, credit hours, etc.). For a full list of content features, refer to [59]. The predicted grade is

$$\hat{g}_{s,c} = w_{c0} + \mathbf{x}_{\mathbf{s,c}}^{\mathbf{T}} \mathbf{w}_{\mathbf{c}}^{\mathbf{f}} \tag{5.14}$$

where $\mathbf{x}_{\mathbf{s,c}}$ is the content feature vector, $\mathbf{w}_{\mathbf{c}}^{\mathbf{f}}$ is the weight vector to be learned and w_{c0} is the bias term.

Course-Specific Hybrid Model

The course-specific hybrid model (CSR_{HY}) predicts a student's grade in a future course by combining the content features and grades of prior courses [59].

5.3 Results and Discussion

5.3.1 Comparative Performance

Table 5.3: Comparative Performance of different models using MAE. (↓ is better)

Method	Fall 2016					Spring 2017				
	CS	ECE	BIOL	PSYC	CEIE	CS	ECE	BIOL	PSYC	CEIE
BO	0.725	0.690	0.541	0.595	0.586	0.763	0.604	0.621	0.609	0.617
MF	0.718	0.679	0.542	0.609	0.579	0.701	0.589	0.625	0.622	0.583
CS_{MF}	0.715	0.666	0.536	0.567	0.573	0.696	0.540	0.624	0.603	0.572
CSR_{PC}	0.680	0.673	0.537	0.493	0.601	0.666	0.506	0.566	0.567	0.517
CSR_{CF}	0.718	0.677	0.476	0.474	0.609	0.683	0.553	0.586	0.565	0.461
CSR_{HY}	0.669	0.663	0.505	0.485	0.583	0.663	0.502	0.560	0.558	0.465
MLP	0.590	0.450	0.429	0.353	0.395	0.606	0.368	0.517	0.491	0.419
LSTM	**0.588**	**0.367**	**0.412**	**0.316**	**0.324**	**0.579**	**0.286**	**0.500**	**0.392**	**0.253**

Table 5.3 shows the comparison of the proposed MLP and LSTM models with various baselines using MAE for the Fall 2016 and Spring 2017 semesters. We observe that the deep learning models have the best performance on all datasets and LSTM outperforms MLP approach. Specifically, the LSTM model outperforms the best performing baseline by 12 to 45% across the different majors and the two semesters. Compared to MLP, the LSTM model is able to achieve better performance. The reason is that LSTM can model the temporal dynamics associated with students' knowledge evolution, which can not be captured by MLP and other traditional methods. In addition, LSTM are able to handle long-term dependencies within the knowledge evolution. For example, an important course such as the prerequisite taken several semesters away can have a significant effect on the course to be predicted, which can be modeled by LSTM.

To gain better insights into the types of errors made by different methods, Table 5.4 presents the experimental results evaluated by using tick errors as defined in Section 5.2.3. The LSTM model achieves the best performance (with exceptions in BIOL and CEIE majors for Fall 2016). Similar to results evaluated by using MAE, MLP is inferior to LSTM but better than the other competing methods. We also observe that the gap between the proposed methods and the baselines is smaller for PTA_2 that allows for errors up to two ticks to be counted as correct. The baseline models with content features show better performance than methods that do not use content features. This suggests that content features are informative for performance prediction.

5.3.2 Identifying At-Risk Students

One of the important applications of student's performance prediction is to develop an early-warning system that can identify students at-risk of failing courses they plan to take in the next term (or future). We define at-risk students as those whose grades are below 2.0 (C and lower). We convert all the grades above 2.0 as not-at-risk and below 2.0 as at-risk, and treat the prediction as a classification problem. The experimental procedures are similar to grade prediction except that the predicted grades over 2.0 are treated as pass and below 2.0 as fail. We choose accuracy and F-1 score as evaluation metrics, due to the fact that the number of at-risk and non-at-risk students is imbalanced as indicated by the percentage of at-risk students at the footnote of Table 5.5 and 5.6. The experimental

Table 5.4: Comparative Performance of Different Models using Tick Error (↑ is better)

		Fall 2016					Spring 2017				
	Method	CS	ECE	BIOL	PSYC	CEIE	CS	ECE	BIOL	PSYC	CEIE
PTA$_0$	BO	6.71	10.38	15.36	27.07	20.08	13.41	13.41	23.22	33.20	28.28
	MF	7.11	13.11	15.90	28.34	21.26	18.49	17.07	23.36	33.21	27.87
	CS_{MF}	6.72	12.57	16.17	28.98	21.26	20.80	14.31	22.38	31.55	31.56
	CSR_{PC}	19.57	20.77	28.84	34.08	27.17	21.87	17.68	25.17	35.19	31.97
	CSR_{CF}	13.44	16.39	28.03	27.39	29.13	15.10	16.78	24.48	33.25	38.52
	CSR_{HY}	19.76	22.40	30.73	35.35	26.38	21.26	18.62	26.15	36.17	37.70
	MLP	26.48	42.62	38.17	46.50	39.40	26.38	42.07	31.61	39.56	33.20
	LSTM	**28.23**	**50.27**	**41.40**	**50.85**	**52.81**	**28.88**	**53.05**	**35.92**	**45.36**	**54.92**
PTA$_1$	BO	29.84	33.33	28.84	45.54	35.83	48.84	43.29	46.79	60.81	70.49
	MF	29.84	31.15	29.65	43.95	34.25	48.69	41.46	47.69	60.80	68.03
	CS_{MF}	30.24	33.87	29.38	45.86	34.65	47.46	40.85	48.95	61.57	70.08
	CSR_{PC}	48.22	55.19	62.80	61.15	52.76	42.84	37.19	46.01	62.37	67.21
	CSR_{CF}	44.66	51.37	70.89	64.97	52.76	45.76	36.59	49.59	65.29	66.39
	CSR_{HY}	49.80	55.19	67.38	61.78	53.15	42.68	36.58	49.73	61.89	66.80
	MLP	57.51	67.76	72.31	72.29	70.99	59.87	73.17	65.03	64.56	65.98
	LSTM	**58.05**	**78.69**	**73.66**	**77.97**	**78.79**	**60.71**	**78.05**	**67.04**	**73.43**	**84.84**
PTA$_2$	BO	60.67	58.47	62.80	69.75	58.66	66.56	62.80	60.22	78.54	82.79
	MF	59.88	57.92	61.46	68.15	57.48	67.79	60.36	61.70	78.64	83.19
	CS_{MF}	61.26	59.02	61.19	71.97	57.48	67.80	60.36	61.79	80.97	83.61
	CSR_{PC}	74.31	73.22	81.40	79.62	69.69	65.02	57.31	63.98	77.18	84.02
	CSR_{CF}	73.52	75.96	87.87	83.44	66.14	65.95	56.71	62.54	78.91	80.10
	CSR_{HY}	75.10	74.32	82.75	78.66	69.29	63.64	57.31	63.84	77.69	81.97
	MLP	79.25	83.61	**87.90**	86.31	**90.91**	78.99	90.24	82.66	80.34	86.48
	LSTM	**79.32**	**88.52**	87.63	**87.80**	88.74	**79.97**	**92.07**	**83.66**	**86.97**	**92.62**

results are shown in Table 5.5 and 5.6 for Fall 2016 and Spring 2017, respectively. Higher value is better. We observe that the proposed methods outperform all the baselines and in most cases, LSTM performs better than MLP.

5.3.3 Uncertainty Evaluation

In this section, we evaluate the quality of our uncertainty estimation in several aspects. The first one is coverage, which can be evaluated by using calibration plot. The calibration plot can be explained by observing that if a prediction is made at 95% confidence level, then the probability that the prediction falls in the prediction interval should be 0.95. Figure 5.2 shows the calibration

Table 5.5: Predictive Power at Identifying At-Risk Students for Fall 2016 (↑ is better)

Method	CS		ECE		BIOL		PSYC		CEIE	
	Acc	F-1	Acc	F-1	Acc	F-1	Acc	F-1	Acc	F-1
BO	0.677	0.275	0.677	0.233	0.776	0.484	0.853	0.041	0.853	0.041
MF	0.703	0.395	0.743	0.483	0.805	0.586	0.863	0.218	0.752	0.350
CS_{MF}	0.774	0.352	0.765	0.188	0.841	0.512	0.863	0.156	0.748	0.333
CSR_{PC}	0.820	0.556	0.786	0.450	0.843	0.539	0.917	0	0.714	0.521
CSR_{CF}	0.796	0.424	0.770	0.192	0.849	0.541	0.942	0.357	0.714	0.388
CSR_{HY}	0.806	0.533	0.792	0.441	0.849	0.562	0.920	0	0.753	0.612
MLP	**0.851**	**0.623**	0.863	0.657	0.865	**0.647**	0.939	0.424	0.870	0.464
LSTM	0.821	0.476	**0.885**	**0.704**	**0.870**	0.625	**0.955**	**0.606**	**0.913**	**0.629**

In Fall 2016, the percentage of at-risk students for each major is CS (24.40%), ECE (19.14%), BIOL (18.79%), PSYC (9.80%), CEIE (12.97%).

Table 5.6: Predictive Power at Identifying At-Risk Students for Spring 2017 (↑ is better)

Method	CS		ECE		BIOL		PSYC		CEIE	
	Acc	F-1	Acc	F-1	Acc	F-1	Acc	F-1	Acc	F-1
BO	0.751	0.320	0.798	0.547	0.814	0.616	0.859	0.194	0.938	0.651
MF	0.730	0.358	0.780	0.571	0.805	0.621	0.854	0.189	0.893	0.458
CS_{MF}	0.779	0.243	0.835	0.542	0.807	0.568	0.873	0.133	0.938	0.482
CSR_{PC}	0.773	0.369	0.823	0.452	0.806	0.560	0.868	0.181	0.889	0.228
CSR_{CF}	0.776	0.248	0.786	0.222	0.805	0.490	0.893	0.120	0.913	0.160
CSR_{HY}	0.779	0.375	0.829	0.517	0.808	0.559	0.864	0.200	0.922	0.296
MLP	0.796	0.388	0.878	0.714	0.830	**0.636**	0.859	0.236	0.938	0.516
LSTM	**0.819**	**0.425**	**0.920**	**0.826**	**0.833**	0.624	**0.907**	**0.274**	**0.954**	**0.666**

In Spring 2017, the percentage of at-risk students for each major is CS (22.19%), ECE (23.78%), BIOL (26.01%), PSYC (10.68%), CEIE (8.61%).

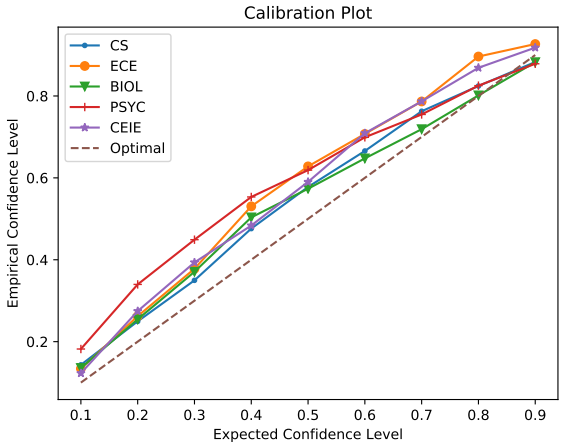


Figure 5.2: Empirical confidence level vs. expected confidence level

plot evaluated on our datasets. The x-axis of the plot is the expected confidence level and the y-axis is the empirical confidence level. From the figure, we can see that the calibration curves across the five majors are close to the optimal calibration curve.

The second way to evaluate uncertainty estimation is that a model should make less errors on predictions that it is confident about. Therefore, we evaluate the model as a function of confidence score and we propose error@k:

$$E@k = \text{error(k most confident predictions)},$$

where error can be mean absolute error or tick error, the predictions are ranked in terms of predictive variance, lower predictive variance is more confident. Figure 5.3 shows mean absolute error with respect to top-k confident predictions. The figure shows that as the predictions become less confident, the prediction errors become higher.

Grade prediction is fundamental for early-warning student facing system. The application case of educational early-warning system is that when an instructor/advisor is informed that a student will fail a course, then the instructor will reach out to the student and provide the student personalized advising and help. In this process, we want to make sure students who are not failing are not predicted as failing students (false positives) so as to reduce wasted educational resources; and

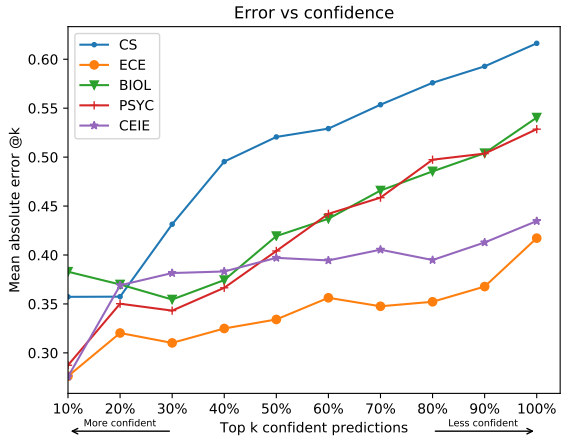
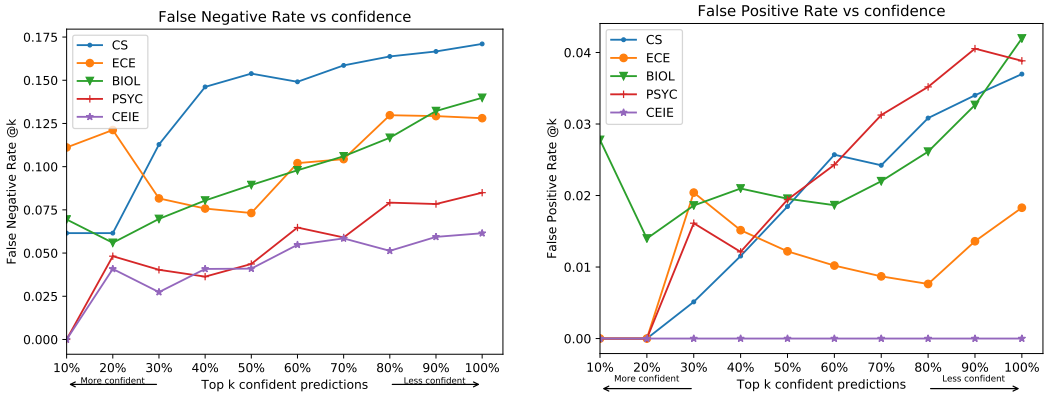


Figure 5.3: MAE as a function of confidence.

the failing students are not predicted as passing students (false negatives) so that they can get much needed timely help. Prediction confidence is useful for reducing these kind of errors by only taking action on confident predictions. We also evaluate uncertainty estimation on the application of identification of at-risk students. Figure 5.4 shows false negative rate (FNR) and false positive rate (FPR) as a function of prediction confidence.



(a) FNR vs confidence

(b) FPR vs confidence

Figure 5.4: FNR and FPR as a function of prediction confidence

We observe in most cases as prediction confidence decreases, there are more errors. To make less errors, we can choose to take actions only on confident predictions. However, more confident

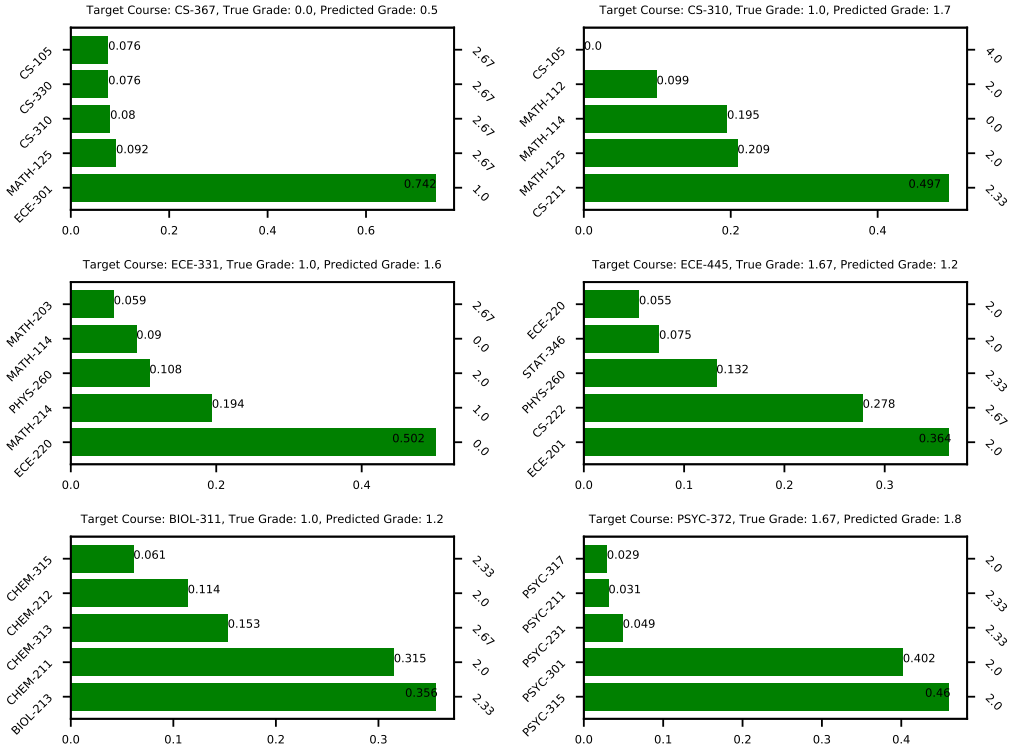


Figure 5.5: Influence of prior courses. For every subfigure, x-axis is influence value, left y-axis is top five most influential prior courses, right y-axis is student's grade in corresponding prior course. In the titles, target course means the course for which we are predicting grade, predicted grade is the predicted grade for the student, and true grade is the student's real grade in the target course.

predictions have less coverage. In practice, we propose to set an appropriate confidence threshold to make a tradeoff between coverage and accuracy.

5.3.4 Case Studies: Influential Courses

To incorporate the developed next-term grade approaches within personalized advising system, we seek to not only report the predicted grades for the student but identify the list of prior courses that were most influential for determining future success in a given target course.

Figure 5.5 shows examples of use case scenarios for students in different disciplines. We choose six at-risk students from CS, ECE, BIOL and PSYCH majors. If a student has grade lower than 2.0, he/she is identified as at-risk student. We compute the influence of the prior courses on the

prediction as described in Section 5.1.5 and only prior courses contributing to the increase in the prediction are reported. The influence is computed by using the LSTM model. The influence index is sorted and normalized and only the top five prior courses are shown in the table. From the top left subfigure, we can see that the student's true grade in class CS-367 (this course is about computer systems and programming) is 0.0, the predicted grade is 0.5 and the most influential prior course is ECE-301 (class about digital electronics) which is a prerequisite. The influence of a prior course is computed by increasing the grade of that course to full grade (i.e. 4.0). This does not suggest that a course with a lower grade has higher influence than a course with higher grade. For example, in top right subfigure, the lowest grade of the student is from MATH-114 (about calculus), but the most influential is course CS-211 (about object-oriented programming) (prerequisite of the target course). The left column of second row shows the third example. The target course is ECE-331 (about digital system design). For this student, the actual grade in the target course is 1.0 and predicted grade is 1.6. The most influential course is ECE-220 (about signals and systems), as this student performed very poorly in this course (grade was 0.0). From the results, we can see that the proposed approach is able to identify influential prior courses that explain the prediction results.

Table 5.7: Top 5 influential prior courses for a target course. Bolded course is the pre-requisite.

Target Course	Top 5 Influential Prior Courses				
CS-367	ECE-301	CS-310	MATH-125	**CS-262**	MATH-213
ECE-433	**ECE-333**	MATH-203	STAT-346	PHYS-160	ENGR-107
PSYC-372	PSYC-325	PSYC-301	PSYC-300	PSYC-211	**PSYC-100**
BIOL-311	BIOL-213	CHEM-211	CHEM-212	CHEM-313	**BIOL-214**
CEIE-331	PHYS-261	**CEIE-210**	STAT-344	PHYS-161	MATH-113

Table 5.7 shows the influence of prior courses on a target course for groups of students. We choose five representative courses from each major. From the table, we observe that for all the courses their prerequisites are one of the top five most influential courses. Although, most of the time, the most influential prior course for a target course is not necessarily the prerequisite, the most influential course is very relevant to the target course. For example, course CS-367 about

low-level computer system such as machine-level programming; the most influential course ECE-301 is digital electronics, which is about designing logic circuits and relevant to low-level computer system. Providing a list of influential courses for a target course can help stakeholders improve the curriculum and program structure.

5.4 Conclusions

In this work, we proposed two course-specific Bayesian deep learning models for next-term grade prediction. The first is a multi-layer perceptron which treats the feature vector as static and ignores the temporal dynamics of a student's knowledge evolution. To overcome this issue, we also developed a Long Short Term Memory model that takes into account the sequential aspect of student's knowledge accumulating process by taking courses across semester/terms.

We highlight the strengths of our proposed approach by incorporating the predictions within three application scenarios: (i) identify at-risk students and (ii) provide explainable results so as to identify a list of influential courses associated with a target course. (iii) provide prediction uncertainty for building a reliable educational early warning system.

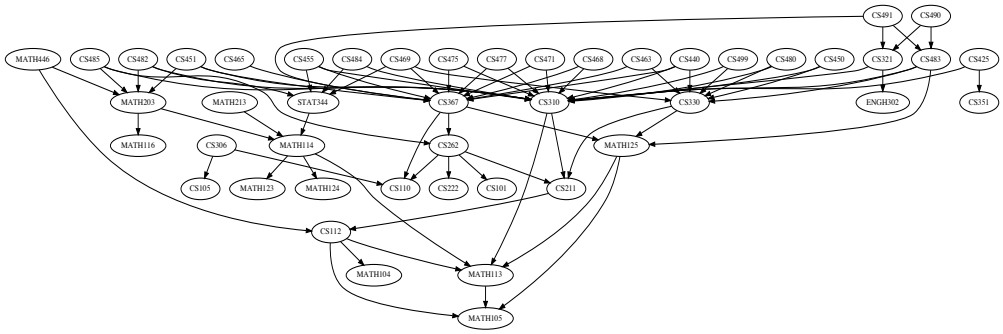
We conducted comprehensive experiments to evaluate the proposed models. The experiments demonstrate that the proposed models exhibit better performance at predicting students' grades than state-of-the-art baselines. The experiments also show that the proposed models have better capability at identifying at-risk students.

Chapter 6: Academic Performance Estimation with Attention-based Graph Convolutional Networks

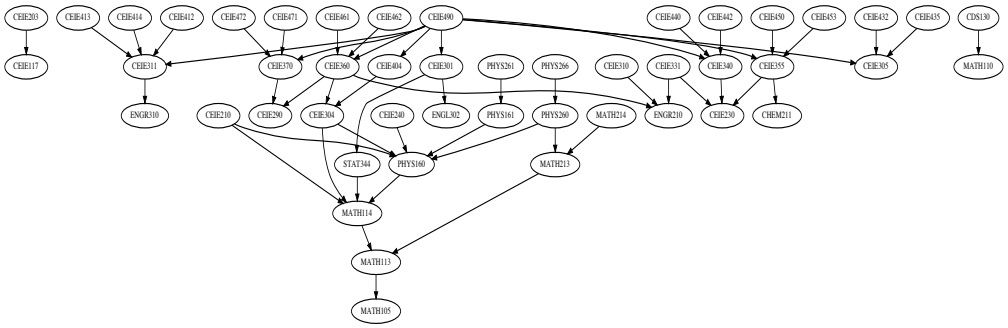
Higher educational institutions face major challenges including timely graduation and retention of enrolled students. The National Center for Education Statistics (NCES) reports that the six-year graduation rate for first-time and full-time undergraduates is around 60%; the retention rate among first-time and full-time degree-seeking students is around 80% [69]. These alarming statistics require higher educational institutions to take actions to improve their effectiveness and efficiency at educating students. Machine learning techniques have been increasingly developed and applied to educational settings in the hope of improving students' learning and increasing students' success [70–72]. Many systems and applications have been proposed; such as course recommender systems [21], academic trajectory and degree planning [73], educational early advising systems [59], and knowledge tracing for intelligent tutoring systems [19, 42]. Developing methods for accurate modeling and predicting students' performance is the key to these systems and applications.

Traditional performance prediction methods can be categorized into two types. The first builds a static model, which takes a feature vector as input (such as a student's grades in previous courses or student-related features) and outputs the predicted grades. A common approach that belongs to this category is linear regression methods [5]. Students take courses sequentially, i.e., they take some courses at each semester; and their performance in courses taken in the next semester depends on courses taken in previous semesters. Further, their knowledge evolves by taking a sequence of courses. To capture the temporal dynamics of students' knowledge evolution, sequential models have been proposed. A set of representative approaches within this category use recurrent neural networks (RNN) [74, 75].

Undergraduate degree programs are designed in a way that knowledge acquired in prior courses serves as prerequisites for future courses. The knowledge and skills required to do well in a course are acquired in multiple prior courses. The knowledge dependence between courses exhibit complex

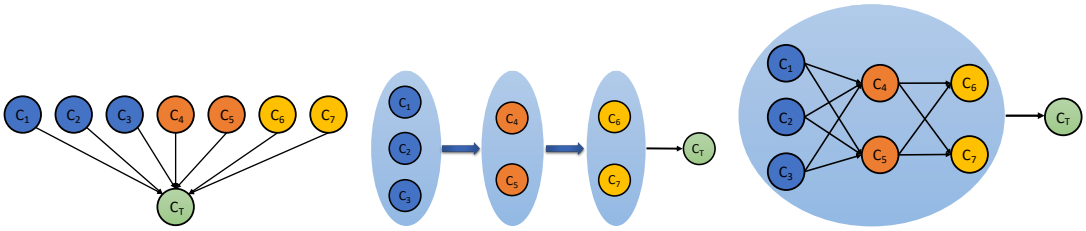


(a) Computer Science



(b) Civil Engineering

Figure 6.1: Course dependence structure in two representative majors.



(a) Static Model (b) Sequential Model (c) Graph Model

Figure 6.2: Comparison of Three Types of Model Architectures. In this example, a student takes course C_1, C_2, C_3 in the first semester, course C_4, C_5 in the second semester, course C_6, C_7 in the third semester before takes the target course C_T.

graph structure as shown in Figure 6.1. Figure 6.1 shows the prerequisite structures for computer science and civil and infrastructure engineering degree programs at George Mason University. Each node represents a particular course. An edge pointing from one course to another shows the prerequisite relationship. As an example, to do well in the data structure course (CS310), students need to acquire programming skills, object-oriented programming knowledge (CS211) and math (MATH113) which come from multiple different courses. The graph in Figure 6.1 also shows hierarchical relationships where a course can depend on another course which is at a much lower academic level. In addition to the prerequisite structures, degree programs are flexible, i.e., students can choose to take elective courses based on their interests and do not have to follow a specific ordering when taking these courses.

The complexity and flexibility of the degree programs make predicting students' performance a challenge task. Prior approaches usually simplify or ignore these complex dependencies. Figure 6.2 shows the comparison of three types of models. Figure 6.2a shows a static model, where a student's performance is directly dependent on a set of prior courses. Figure 6.2b shows a sequential model, where students' knowledge evolution is partially modeled. To overcome the constraints and limitations of the traditional models, we propose a model based on graph convolutional networks to capture the complex graph-structured knowledge evolution exhibited by students' data. Specifically, we propose an attention-based graph convolutional network (ACGN) model for predicting a student's grade in a future course. Figure 6.2c shows the graph model, where each course depends

on all courses taken in the semester before it so that students' knowledge evolution is fully captured.

When a system is used for decision making e.g., as a support tool for advisors to identify students who are at-risk of failing courses they will take; it is essential for the predictions to be interpretable. This allows the stakeholders to trust the decision making systems and make informed decisions. We show that our attention-based model is able to provide an interpretable and useful explanation for the predictions. Our model is able to analyze a student's performance in prior courses and identify a collection of important prior courses to explain the student's performance in target course.

We performed extensive experiments on real-world datasets to evaluate our model and compare it with the other two types of models aforementioned. The experimental results are consistent with our observations that models with architectures more close to the degree program have better modeling capability and prediction performance. One of the important applications for students' performance prediction is early warning and advising systems, where at-risk students are first identified and timely support is provided to improve their academic success. The experimental results show our model's effectiveness at identifying at-risk students.

The key contributions of the paper are summarized as follows:

- Flexible graph structured model for students' academic performance prediction. Observing the complex structures of undergraduate degree programs, we propose a graph convolutional network model for students' performance prediction.

- Attention based model for explanation. Providing explanations for a model's predictions makes the model useful for decision making. Our attention-based model can explain the predictions by identifying a set of prior courses important for the predictions.

- Identification of at-risk students. While most models achieve good performance at predicting students' performance, they suffer from low accuracy at identifying at-risk students. Our proposed model is able to achieve comparable performance with state-of-the-art models.

6.1 Related Work

The need to improve higher education services and offerings has attracted research on developing methods for predicting students' performance [76, 77]. In this section, we review related work on students' performance prediction. The related work can be classified into three categories: (i) static models, (ii) sequential models and (iii) graph models.

6.1.1 Static Models

Static grade prediction models learn a mapping function, where input is student-related features and the output is predicted grade. Polyzou et al. [5] proposed regression models specific to courses or students for predicting a student's grade in a target course. They found that focusing on a course specific subset of the data leads to more accurate predictions. Elbadrawy et al. [78] introduced a personalized multi-regression model for predicting students' performance in course activities. Compared to a single regression model, this model is able to capture personal student differences. To understand how students' behavior impacts their academic performance, Wang et al. [79] collects students' behavioral data using smart phone for performance prediction. Many other classic supervised learning approaches have been used for students' performance prediction including decision trees [24], support vector machines and neural networks [25].

Adapted from recommender systems domain, matrix factorization [27] approaches are popular for grade prediction. These factorization approaches make the assumption that a student's knowledge/skills and a course's knowledge components can be jointly represented with latent vectors (factors) [4]. Polyzou et al. [5] proposed course-specific matrix factorization models for grade prediction that decompose a course-specific subset of students' grade data. The student course records also exhibit grouping structures and a domain-aware matrix factorization model was developed for the joint course recommendation and grade prediction [21]. Ren et al. [29] proposed matrix factorization model coupled with temporal dynamics for grade prediction.

6.1.2 Sequential Models

Students take courses sequentially. Their knowledge and skills evolve by taking a series of courses. To model the temporal dynamics of students' knowledge evolution, sequential models have been proposed. Balakrishnan [80] proposed a Hidden Markov Model for predicting student dropout by modeling students' activities over time in a Massive Open Online Courses (MOOCs). Swamy et al. [81] models student progress on coding assignments in large-scale computer science courses using recurrent neural networks. Kim et al. [74] proposed a bidirectional long short term memory (BLSTM) model for the online educational setting. Hu et al. proposed course-specific markovian models for students' grade predictions [82]. Morsy et al. proposed cumulative knowledge-based regression models for next-term grade prediction, which models students' knowledge evolution by using a sequential regression model. Hu et al. [75] proposed long short term memory models for grade prediction in traditional higher education.

6.1.3 Graph Neural Networks Models

Deep learning approaches have found unprecedented success in a myriad of applications involving regular structured data such as images (grids) and text (sequences) [43]. Graphs are more complex and irregular than grids or sequences and recent research efforts involve designing deep learning models for graph data. Graph neural networks have been proposed and applied to many areas such as computer vision for point clouds classification [83], action recognition [84]; recommender systems [85] and traffic prediction [86]. To the best of our knowledge, there is no prior work on students' performance prediction using graph neural networks.

6.2 Methods

6.2.1 Problem Statement

Given a student s, the set of courses taken and grades obtained in term t are represented by $\mathbb{P}_s^t$. For a sequence of terms $1 \dots T_s$, we denote $\mathbb{P}_s^{1 \sim T_s} = \mathbb{P}_s^1, \mathbb{P}_s^2, \dots, \mathbb{P}_s^{T_s}$ to represent the sequence of courses taken and grades obtained by student s in T_s terms. For a target course c taken in the future (next)

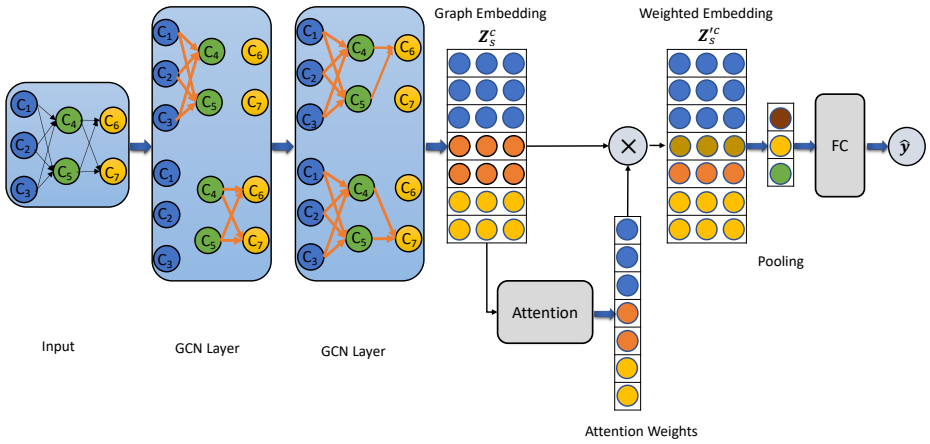


Figure 6.3: The proposed model.

term, the objective of the proposed method is to predict the grade student s will achieve in course c denoted by $\hat{g}_s^c$.

The proposed models are trained in a course specific manner i.e., for each target course c we learn a unique model. Due to the flexibility of academic degree programs, in each semester different courses can be taken; and for each student, the number of semesters studied before taking the target course will be different. Therefore, we index the length of the sequence with student-specific variable T_s.

For every target course c, a subset of frequently taken prior courses are identified from all the prior courses taken by students who have already taken the target course c. These prior courses are denoted as $\mathbb{C}_c$ of size N_c. For student s, only the prior courses in $\mathbb{C}_c$ are extracted from $\mathbb{P}_s^{1\sim T_s}$ to form a graph which is represented by an adjacency matrix $\mathbf{A}_s^c \in \{1,0\}^{N_c \times N_c}$ and a feature matrix $\mathbf{F}_s^c \in \mathbb{R}^{N_c \times D}$, where D represents the number of features. Take Figure 6.2c as an example, the student takes courses c_1, c_2, c_3 in the first term, c_4, c_5 in the second term and c_6, c_7 in the third term; we want to predict his/her grade in course C_T. Adjacency matrix $\mathbf{A}_s^c$ for this student represents his course taken process. Courses taken in the current term are fully connected to courses taken in the next term; 1 represents connected, 0 otherwise. A row of the feature matrix $\mathbf{F}_s^c$ represents the student's grades in corresponding prior courses.

6.2.2 Model Description

Figure 6.3 shows an overview of the proposed model. It is composed of three parts: 1) graph convolutional network, 2) attention layer and 3) a fully connected layer.

Graph Convolutional Network (GCN)

Convolutional neural networks (CNNs) show superior performance on several applications related to vision [87], speech and text [88]. CNNs are powerful because of their ability to exploit feature locality at multiple granularity. Graph Convolutional networks have a similar working mechanism but on data with more complex structures, namely, graph.

The input to a GCN is an adjacency matrix $\mathbf{A}_s^c$ and feature matrix $\mathbf{F}_s^c$, encoding student s's course taking process and grades in prior courses, respectively. Multiple layers of graph convolutional layer are applied on $\mathbf{A}_s^c$ and $\mathbf{F}_s^c$ to learn a graph level embedding $\mathbf{Z}_s^c \in \mathbb{R}^{N \times D}$. Each row of $\mathbf{Z}_s^c$ corresponds to a node embedding vector. A graph convolutional layer is mathematically described as follows:

$$\mathbf{H}^{(l+1)} = f(\mathbf{H}^{(l)}, \mathbf{A}) = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}) \tag{6.1}$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ is the adjacency matrix with self-connections, $\tilde{\mathbf{D}} = \Sigma_j \tilde{\mathbf{A}}_j$ is the normalization matrix, $\mathbf{H}^{(l)}$ is the input and $\mathbf{W}^{(l)}$ is the weight matrix to be learned. $\mathbf{H}^{(0)} = \mathbf{F}_s^c$ and $\mathbf{H}^{(L)} = \mathbf{Z}_s^c$; namely, the input into the first GCN layer is the feature matrix $\mathbf{F}_s^c$, the output from the last GCN layer is the student-specific graph embedding $\mathbf{Z}_s^c$.

A filter in convolutional neural networks aggregates information from a pixel's neighbors. Similarly, the graph convolutional layer aggregates information from a node's neighboring nodes and generates a new node embedding vector by the following equation

$$\mathbf{h}_i = \sigma(\Sigma_j \frac{1}{c_{ij}} \mathbf{h}_j \mathbf{W}) \tag{6.2}$$

where node j is node i's neighbor. A higher level of the node embeddings are generated by applying multiple GCN layers. Multiple layers of GCN aggregate information from a node's further neighbors. As shown in Figure 6.3, the first GCN layer aggregates information from a node's direct neighbors, namely, in our case the courses taken in last semester. The second layer collects information from a node's second degree neighbors, i.e., the courses taken two semesters ago. The final output is the graph embedding which entails information from all the courses a student has taken.

Attention Layer

The output from GCNs is a graph-level embedding matrix, which encodes information about a student's knowledge and skills acquired in prior courses. The knowledge acquired from different prior courses has different importance for the target course. To capture the importance differences of the prior courses, we integrate attention layer into our model. Attention mechanism allows the model to focus on the relevant features or information useful for prediction. It works by computing an importance score [89], higher score means the corresponding prior course is more important for predicting a student's performance; given by

$$e_i = MLP(\mathbf{h}_i) \tag{6.3}$$

$$\alpha_i = \frac{\exp(e_i)}{\Sigma_{k=1}^{N} \exp(e_k)} \tag{6.4}$$

where MLP is a learnable function, i.e., multi-layer perceptron, α_i is the attention score corresponds to $\mathbf{h}_i$. The output from the attention layer is an attention score vector $\boldsymbol{\alpha}$.

The graph embedding matrix $\mathbf{Z}_s^c$ is weighted by attention scores to form a weighted graph embedding matrix $\mathbf{Z}_s^{'c}$ given by

$$\mathbf{Z}_s^{'c} = \begin{bmatrix} \alpha_1 \mathbf{z}_{s,1}^c \\ \vdots \\ \alpha_i \mathbf{z}_{s,i}^c \\ \vdots \\ \alpha_N \mathbf{z}_{s,N}^c \end{bmatrix} \tag{6.5}$$

Finally, the pooling layer coarsens the weighted graph embedding matrix into a latent vector $\mathbf{v}_s^c$. The latent vector is passed through a multilayer perceptron; the output from which is the predicted grade.

$$\hat{g}_s^c = f(\mathbf{v}_s^c) \tag{6.6}$$

where f is a multilayer perceptron network.

6.3 Experimental Protocol

6.3.1 Dataset Description

Table 6.1: Dataset Statistics

Major	Fall 2017			Spring 2018		
	#S	#C	#G	#S	#C	#G
CS	5,042	16	47,889	5,297	20	52,152
ECE	1,992	18	34,355	1,980	18	34,170
BIOL	7,065	20	52,574	6,976	20	52,672
PSYC	5,367	20	25,207	5,368	20	25,247
CEIE	2,222	17	30,956	2,181	16	30,283
Overall	21,688	91	190,981	21,802	94	194,524

#S total number of students, #C number of courses for prediction, #G total number of grades

The data is collected at George Mason University from Fall 2009 to Spring 2018. The five largest majors are chosen including: 1) Computer Science (CS), 2) Electrical and Computer Engineering (ECE), 3) Biology (BIOL), 4) Psychology (PSYC) 5) Civil Engineering (CEIE). The evaluation procedure is designed in a way to simulate the real-world scenario of predicting the next-term grades. Specifically, the models are trained on the data up to term $T-2$ and validated on term $T-1$ and tested on term T. The latest two terms are chosen as testing terms, i.e. term Fall 2017 and term Spring 2018. For example, to evaluate the performance of the models on term Fall 2017, the model is trained on data from term Fall 2009 to term Fall 2016, validated on term Spring 2017 to choose the parameters associated with different approaches and finally tested in term Fall 2017. The statistics of the datasets are listed in Table 7.1

6.3.2 Evaluation Metrics

We evaluate the models from two perspectives: 1) the accuracy of grade predictions, 2) the models' ability at detecting at-risk students.

To evaluate the models' accuracy of grade prediction, two evaluation metrics are used a) mean absolute error (MAE) and b) percentage of tick accuracy (PTA).

$$\text{MAE} = \frac{\sum_{i=1}^{N} |g_i - \hat{g}_i|}{N} \tag{6.7}$$

where g_i is true grade and $\hat{g}_i$ is predicted grade.

In the grading system, there are 11 letter grades (A+, A, A-, B+, B, B-, C+, C, C-, D, F) which correspond to (4, 4, 3.67, 3.33, 3, 2.67, 2.33, 2, 1.67, 1, 0). A tick is the difference between two consecutive letter grades. The performance of a model is estimated by how many ticks away the predicted grade is from the true grade. For example, the tick error between B and B is zero, B and B+ is one, B and A- is two. To use PTA for evaluation, we first convert the predicted numerical grade to its closest letter grade and then compute the percentage of errors with 0 tick, within 1 tick, and within 2 ticks denoted by PTA_0, PTA_1, and PTA_3, respectively.

We also evaluate the models' performance of identifying at-risk students. At-risk students are

defined as those whose grades are lower than 2.0 (C, C-, D, F). The predicted grades below 2.0 are treated as positives and above 2.0 are treated as negatives. The process of detecting at-risk students is similar to grade prediction except that the output from the model (the predicted grade) is converted to 1 or 0 based on whether the predicted grade is below or above 2.0. As the number of at-risk students is low, we use F-1 score as evaluation metric.

6.3.3 Comparative Methods

Bias Only (BO)

Bias only method only takes into account a student's bias, a course's bias and global bias[5]. The predicted grade is as follow

$$\hat{g}_s^c = b^c + b_s^c + b_{c'}^c \tag{6.8}$$

where $b^c, b_s^c, b_{c'}^c$ are global bias, student bias and course bias, respectively.

Course Specific Matrix Factorization (CSMF)

The key assumption underlying this model is that students and courses can be jointly represented by low-dimensional latent factors. N, M and D is the number of students, courses and latent dimension, respectively [5]. To predict a student's grade in a course, we have:

$$\hat{g}_s^c = b^c + b_s^c + b_{c'}^c + < \mathbf{u}_s^c, \mathbf{v}_{c'}^c > \tag{6.9}$$

where b^c is global bias, b_s^c is student bias term, $b_{c'}^c$ is course bias term; $\mathbf{u}_s^c$ is student s's latent vector, $\mathbf{v}_{c'}^c$ is course c's latent vector.

Course Specific Regression (CSR)

Course specific regression (CSR) [5] is a linear regression model. The input into this model is a vector $\mathbf{x}_s^c$ representing a student's grades in prior courses. A course specific subset of prior courses

included in $\mathbb{P}_s^{1\sim T_s}$ are flattened to form the vector $\mathbf{x}_s^c$. The predicted grade is

$$\hat{g}_s^c = w_0^c + \mathbf{x}_s^c \mathbf{w}^c \tag{6.10}$$

where w_0^c is bias term and $\mathbf{w}^c$ are weight vectors to be learned.

Multilayer Perceptron (MLP)

Multilayer Perceptron is a generalized version of CSR. CSR model is a linear model, which is not able to capture non-linear and complex patterns in students' grades data. Therefore, multilayer perceptron has been proposed by [75] for grade prediction. Similar to CSR, the input $\mathbf{x}_s^c$ is a student's grades in prior courses.

$$\hat{g}_s^c = f(\mathbf{x}_s^c) \tag{6.11}$$

where f is the model to be learned.

Long Short Term Memory (LSTM)

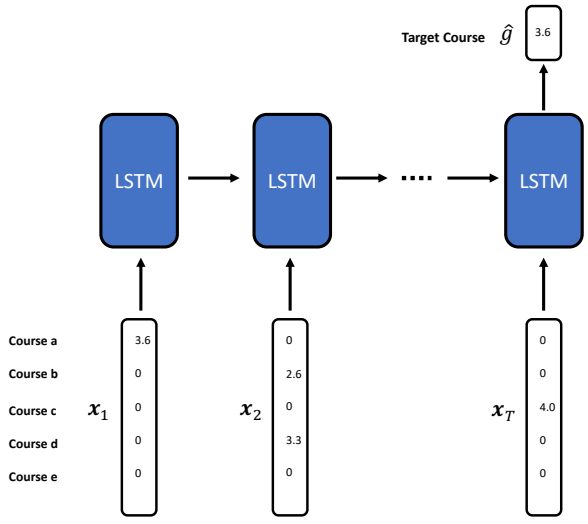


Figure 6.4: LSTM for grade prediction

Long short term memory (LSTM) is an extension of recurrent neural networks (RNN) for modeling sequential data. The assumption of using LSTM for students' performance prediction is that students knowledge and skills are evolving by taking courses in each semester. To capture the temporal dynamics of students' knowledge evolution, LSTMs have been proposed in [75]. The input $\mathbf{x}_{s,t}^{c}$ at time step t is a student's grades in courses at semester t. Many to one architecture is utilized and the output from the last step of LSTM is fed into a fully connected network; the output from which is the predicted grade. The model architecture is shown in Figure 6.4, where the courses a, b, c, d, e are prior courses, $\mathbf{x}_{s,t}^{c}$ encodes the student's grades in courses at time t and the output $\hat{g}$ is the predicted grade.

6.3.4 Implementation

Our method is implemented in Pytorch [90]. For model optimization we use Adam [68]. To avoid model overfitting, we used l_2 norm regularization (with coefficient 0.001) and dropout (dropout rate 0.05) [91]. The number of dimensions for the graph embedding is chosen from a list of (8, 12, 16, 20, 32, 64).

6.4 Experimental Results

6.4.1 Grade Prediction

Table 6.2: Comparative Performance of Different Models by MAE. (↓ is better)

Method	Fall 2017					Spring 2018				
	CS	ECE	BIOL	PSYC	CEIE	CS	ECE	BIOL	PSYC	CEIE
BO	0.684	0.570	0.705	0.556	0.616	0.727	0.674	0.628	0.552	0.605
CSMF	0.594	0.476	0.550	0.517	0.479	0.647	0.539	0.499	0.492	0.491
CSR	0.607	0.444	0.551	0.440	0.441	0.628	0.493	0.463	0.439	0.444
MLP	0.585	0.390	0.515	0.407	0.413	0.590	0.436	0.417	0.413	0.369
LSTM	0.582	0.365	0.532	0.380	**0.309**	0.590	0.370	0.435	0.356	**0.251**
AGCN	**0.540**	**0.335**	**0.459**	**0.309**	0.336	**0.543**	**0.366**	**0.379**	**0.316**	0.258

Table 6.3: Comparative Performance of Different Models by Percentage of Tick Accuracy (↑ is better)

		Fall 2017					Spring 2018				
	Method	CS	ECE	BIOL	PSYC	CEIE	CS	ECE	BIOL	PSYC	CEIE
$\mathbf{PTA}_0$	BO	16.76	20.75	14.40	15.52	14.90	16.07	12.11	15.42	13.65	19.79
	CSMF	20.00	23.58	22.40	23.10	28.85	22.31	17.37	23.35	28.41	28.65
	CSR	24.26	33.96	27.60	38.97	40.87	26.29	28.42	34.14	41.33	35.42
	MLP	26.32	39.62	31.00	41.72	41.35	27.76	33.68	41.41	43.17	42.19
	LSTM	27.21	**42.92**	37.40	48.62	49.52	30.54	**54.74**	42.73	49.82	**57.29**
	AGCN	**30.00**	41.51	**38.80**	**56.21**	**50.00**	**36.52**	39.47	**44.49**	**50.55**	56.77
$\mathbf{PTA}_1$	BO	44.71	49.06	43.20	57.59	48.56	44.09	37.37	46.70	57.20	50.00
	CSMF	55.15	62.26	60.00	63.10	62.98	52.72	54.74	63.66	59.04	61.98
	CSR	55.29	66.04	59.40	66.21	71.63	57.37	63.68	66.30	65.31	69.27
	MLP	56.91	69.81	62.80	69.66	74.52	60.03	68.42	68.28	69.37	76.04
	LSTM	58.24	73.11	61.40	73.79	79.33	59.10	72.11	72.03	75.65	82.81
	AGCN	**62.21**	**75.47**	**70.00**	**77.93**	**79.81**	**63.61**	**77.89**	**74.89**	**77.86**	**84.90**
$\mathbf{PTA}_2$	BO	72.94	81.13	72.40	84.83	81.25	73.97	74.21	77.75	87.45	79.17
	CSMF	80.00	86.79	83.60	83.45	87.50	75.30	84.21	84.36	85.24	84.38
	CSR	76.76	86.32	80.80	83.45	84.62	77.03	82.63	84.58	82.66	86.46
	MLP	79.85	89.62	82.80	85.86	86.54	79.42	86.32	86.34	84.13	90.62
	LSTM	77.35	86.79	79.20	84.83	90.87	77.69	83.16	84.58	**89.67**	91.67
	AGCN	**81.47**	**92.45**	**85.60**	**88.62**	**91.83**	**80.21**	**88.95**	**87.67**	88.93	**93.23**

Table 6.2 reports the performance of ACGN and comparative approaches for the task of next-term grade prediction for the Fall 2017 and Spring 2018 semesters using the MAE metric. The proposed ACGN model achieves the best performance in most cases except the Civil Engineering (CEIE) major. The CEIE major has relatively simpler knowledge dependence structure as shown in Figure 6.1b. A majority of higher level courses, such as 300 and 400 level courses for the CEIE major have shallow knowledge dependence. While for CS major, the higher level courses have deeper knowledge dependence or longer pre-requisite chains.

Another observation is that models which are able to capture the complex knowledge dependence more have better performance. The static models (BO, CSMF, CSR, MLP) are outperformed by sequential model (LSTM) in most cases, on average by 9.2%; the sequential model is outperformed by graph model (AGCN), besides CEIE major, on average by 7.0%. The experimental

results are consistent with our assumption that the knowledge dependence in the undergraduate degree programs is complexly networked structures and a graph model is well-suited at capturing the underlying dynamics.

Table 6.3 shows the comparative performance using the percentage of tick error accuracy. In contrast to MAE, the PTA metric can provide a fine-grained view of the errors made by different methods. From Table 6.3 we observe that the performance gap between models at $\mathbf{PTA}_0$ is larger than at $\mathbf{PTA}_2$. For example, for CS majors in Fall 2017, the gap between the best performing model AGCN and the worst performing model BO at $\mathbf{PTA}_0$ is 13.24%, which is larger than 8.53% at $\mathbf{PTA}_2$.

Table 6.4: Predictive Performance at Identifying At-risk Students, F-1 Score (↑ is better)

Method	Fall 2017					Spring 2018				
	CS	ECE	BIOL	PSYC	CEIE	CS	ECE	BIOL	PSYC	CEIE
BO	0.092	0.000	0.116	0.000	0.000	0.085	0.000	0.194	0.000	0.000
CSMF	0.385	0.415	0.585	0.154	0.429	0.349	0.291	0.620	0.364	0.526
CSR	0.398	0.514	0.649	**0.438**	0.490	0.500	0.543	0.623	0.429	0.450
MLP	0.383	0.426	0.630	**0.438**	0.500	0.534	0.472	0.676	0.400	0.605
LSTM	0.492	**0.533**	0.553	0.276	**0.702**	0.584	**0.650**	0.638	0.400	**0.681**
AGCN	**0.516**	0.500	**0.660**	**0.438**	0.615	**0.594**	0.571	**0.685**	**0.483**	0.550

The percentage of at-risk students for each major in Fall 2017 is CS (23.7%), ECE (18.9%), BIOL (25.8%), PSYC (8.3%), CEIE (15.9%); In Spring 2018, it is CS (23.7%), ECE (24.7%), BIOL (18.1%), PSYC (6.6%), CEIE (14.1%).

6.4.2 Detecting At-risk Students

Detecting at-risk students early is a fundamental task for early warning and advising systems. We evaluate the models' performance at detecting at-risk students. Table 6.4 shows the experimental results evaluated by F-1 score. The percentages of at-risk students in different majors are presented at the table footnote. The PSYC major has the lowest percentage of at-risk students. The experimental results show that LSTM and AGCN achieve the best performance at detecting at-risk students. BO performs worst at the detection of at-risk students. BO only captures the average performance of

a student and a course, which is biased by other students and courses' performance and the average performance of other students and courses is usually higher than 2.0 (the threshold of defining at-risk students).

6.4.3 Interpretation with Attention

Table 6.5: Case Studies By Attention Score

Target Course	True Grade	Predicted Grade	Prior Courses	Grades	Attention Score
CS-310	F	C-	MATH-213	N	0.33
			MATH-125	N	0.33
			CS-262	N	0.33
			CS-211	C	0.01
CS-310	D	D	MATH-213	F	0.913
			MATH-114	C	0.072
			CS-211	N	0.015
BIOL-311	F	C	**BIOL-213**	C+	0.5315
			BIOL-214	C+	0.4685
BIOL-452	D	C	**CHEM-211**	C+	0.5271
			BIOL-214	B	0.2784
			BIOL-213	C	0.1945

N means that the student did not take the course. Courses in bold mean they are in prerequisites chain.

Machine learning models have achieved impressive performance in many tasks. However, most of them remain black boxes and there are concerns about their transparency. A model's capability to provide explanations for its predictions can increase its transparency. For decision making, understanding the reasons behind predictions can help decision makers make informed decisions. Grade prediction models serve as an assistant tool for advisors to make decisions on whether to intervene on a student or not. When the model predicts that a student is at-risk of failing a course, knowing which prior courses results in the prediction can also help advisors provide personalized feedback to students.

Attention mechanism works by letting the model focus on important information for prediction. In our proposed model, the design of the attention layer lets the model focus on important prior courses. The output from the attention layer is a vector of scores representing the importance of the prior courses computed by Equation 6.4. In this section, we show by case studies how the attention

scores from the attention layer explain the model's predictions, especially, why the model predicts that a student is at-risk of failing a target course.

Table 6.5 shows four case studies. We keep the most important prior courses identified by attention score. For the first case study, the target course is CS-310, the student's true grade in the target course is F and the predicted grade is C-. The most important four courses identified by attention layer is MATH-212, MATH-125, CS-262, CS-211. The reason for predicting this student as at-risk is that the student did not take MATH-212, MATH-125, CS-262, therefore lacks the necessary knowledge to do well in the target course. In the second case, the student's true grade in CS-310 is D, the predicted grade is D. The three most important courses are MATH-213, MATH-114, CS-211. The reason for predicting this student as failing the target course is that he failed MATH-213 and did not do well in MATH-114 and did not take CS-211, which is the prerequisite of the target course. In the third case, the student's true grade in the target course is F and the predicted grade is C. The two most important prior courses identified are BIOL-213 and BIOL-214, both are in prerequisite chain of the target course and the student did not do well in them. The fourth case shows that the student failed the target course BIOL-452 and the predicted grade is C. The three most influential prior courses are CHEM-211, BIOL-214, BIOL-213. Courses CHEM-211 and BIOL-213 are in prerequisite chain and the student did not perform well in them.

From the case studies, we can see that the attention layer identifies missing knowledge components for a target course, arising due to two reasons: 1) the student did not take some important prior courses, 2) the student did not do well in the corresponding prior courses.

6.4.4 Sensitivity Analysis

In this section, we evaluate the sensitivity of the model's performance with respect to the dimension of the graph embedding. In Figure 6.5, the x-axis is the embedding dimension and y-axis is MAE for Fall 2017 and Spring 2018 datasets. From Figure 6.5, we can see that the model's performance varies with the dimension size. Overall, its performance is quite stable across the different majors.

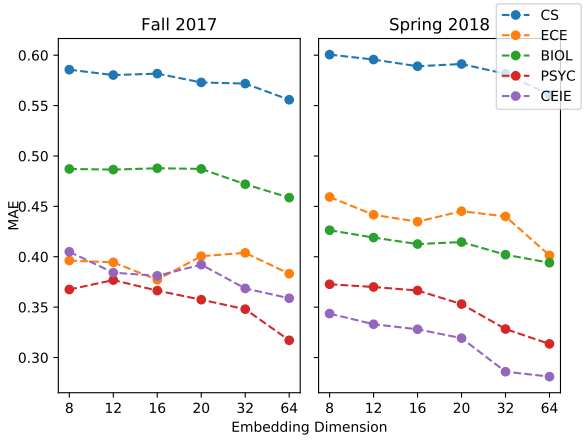


Figure 6.5: Sensitivity analysis on embedding dimension.

6.5 Conclusions

Students' performance prediction is a fundamental task in educational data mining. Predicting students' performance in undergraduate degree programs is a challenging task due to several reasons. First of all, undergraduate degree programs exhibit complex knowledge dependence structures. Secondly, undergraduate degree programs are flexible which means students can take courses without following specific order and they can choose to take whatever electives they are interested in. Traditional approaches like static and sequential models are not able to fully capture the complexity and flexibility of students' data.

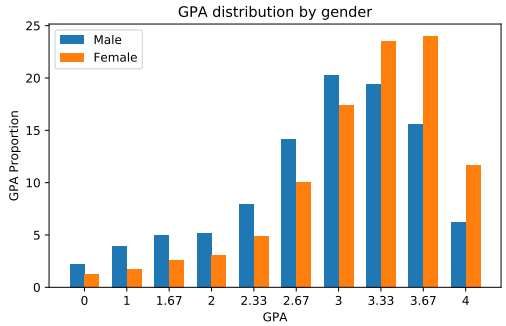
In this work, we proposed a novel attention-based graph convolutional networks for students' performance prediction. The model is able to capture the relational structure underlying students' course records data. We performed extensive experiments to evaluate the proposed model on real-world datasets. The model is evaluated in several aspects: 1) grade prediction accuracy and 2) ability to detect at-risk students. The experimental results show that our model outperformed state-of-the-art approaches in terms of both grade prediction accuracy and at-risk students detection. Finally, the attention layer provides explanations for the model's prediction, which is essential for decision making.

Chapter 7: Towards Fair Educational Data Mining: A Case Study on Detecting At-risk Students

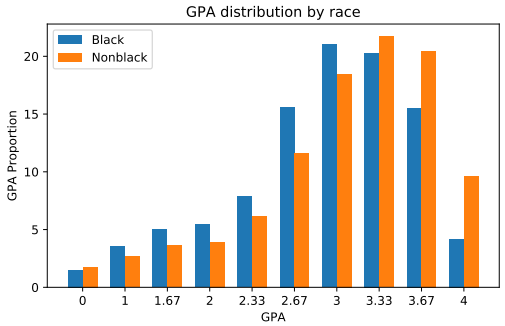
Educational data mining (EDM) approaches seek to analyze student-related data with the objective of improving learning outcomes for students. Many machine learning methods have been proposed for student modeling and forecasting. However, in the past few years, concerns and evidence have emerged about the fairness of machine learning models. A investigation by ProPublia has found that a machine learning tool COMPAS used to predict risk of recidivism exhibits alarming bias against African-American defendants. It shows that the false positive rate of African-American defendants is twice as their white counterparts (45% vs. 23%) [12]. Buolamwini et al. [15] observed imbalanced gender and skin type distributions in facial recognition datasets. Their study show that facial recognition algorithms are more likely to misclassify darker-skinned females with error rates up to 34.7%, while the maximum error rate for light-skinned male is 0.8%. In health care, an algorithm used to guide health decisions are found that African-American patients assigned the same level of risk are sicker than white patients [13].

In the domain of EDM, unfairness has also been observed. In academic performance prediction systems, social indicators have been found to predict low-performance of male students more accurately than that of female students [92]. A study by Doroudi et al. [93] showed that although personalized models were more equitable than treating all students the same, they were still not fair when relying on inaccurate models and the inequities could cascade as the amount of content increases.

Machine learning models learn from data. If bias is recorded in data, models trained on the biased data can also be biased [15]. Bias is also observed in educational data. Figures 7.1a and 7.1b show the average GPA of students by gender and race at University X over a period of ten years. The GPA of a student is his/her accumulative GPA as of the last term before graduation. In Figure 7.1a, average GPA of male students is skewed towards lower GPAs, while average GPA of female



(a) GPA distribution by gender.



(b) GPA distribution by race.

students is skewed towards higher GPAs. The average GPA of overall female students is 3.15 which is higher than that of male students 2.86. Figure 7.1b shows the average GPA of African-American and non-African-American students. From the figure, we can observe that average GPA of African-American students leans towards left while that of non-African-American students leans towards right. The data shows that the average GPA of African-American students is 2.86, while it is 3.03 for non-African-American students.

Biased data can lead to biased machine learning models which can be harmful to minority groups. For example, in education models predicting a group of students to be at-risk or underperforming can discourage them and undermine their learning outcomes. To resolve the harmful results brought about by inequity of machine learning, there are critical needs to develop fair machine learning algorithms.

Machine learning community has been working on formalizations and evaluation of fairness. Different concepts of fairness have been proposed such as statistical parity [94], equalized odds and equal opportunity [95], counterfactual fairness [96] and individual fairness [97]. The above-mentioned fairness definitions can be categorized into two types: group fairness and individual fairness. Group fairness focus fairness on group level, which might not be fair to some individuals in a particular group. Individual fairness on the other hand requires that individuals are treated equally, which is more desirable [97]. However, traditional individual fairness is based on the idea that similar individuals should be treated similarly, which relies upon a problem-specific similarity

metric. Its reliance on similarity metric makes it hard to adapt to different tasks. Metric-free individual fairness has been proposed [98], which eliminates the requirement of a similarity metric. In this work, we develop a fair model based on metric-free individual fairness.

Metric-free individual fairness assumes that an individual's qualification should not be changed if his/her sensitive attribute is changed. In this paper, without loss of generality we assume there are two sensitive attributes. The proposed model is composed of two classifiers. Each classifier corresponds to a sensitive group. The classifier corresponding to the individual's sensitive attribute predicts the individual's probability of being positive, while the probability of the other classifier indicates the individual's probability of being positive if his/her sensitive attribute is changed. According to the definition of metric-free individual fairness, the two probability distributions should be nearly identical. The proximity of the two probability distributions is treated as fairness. The closer the two distributions, the fair the prediction is. In addition to fairness, we also care about the accuracy of the classifier. Therefore, the overall objective we seek to optimize is the accuracy of the classifier corresponding to the individual and the proximity of the distributions of the two classifiers. We also explore using contextual bandits as builing block and propose an algorith called cooperative contextual bandits. The algorithm consists of multiple contextual bandits where each bandit corresponds to a sensitive attribute. Given an individual, the algorithm predicts the individual's probability of getting a positive outcome using bandit corresponds to his/her sensitive attribute and compares it to the probability of getting a positive outcome by changing his/her sensitive attribute. The proximity of the two probability distributions is treated as fairness. Namely, if the qualification of the individual is not affected by his sensitive attribute, it is considered fair. The algorithm treats the proximity of the two probability distributions as a reward signal and maximizes it to achieve fairness. As we need to measure the divergence of the predictions between different bandits, we develop a gradient contextual bandits algorithm. Compared to traditional contextual bandits which predict the action value, gradient contextual bandits directly map a state to a action distribution that can be used to measure divergence. The advantage of this algorithm is that the fairness is treated as reward and the fairness criterion does not need to be differentiable.

The proposed models are evaluated on datasets collected from University X and the task is detecting at-risk students. The experimental results show the efficacy of the proposed models at mitigating bias. Although, the overall data shows that female and non-African-American students have higher overall performance, we observe that the bias is different for different courses. Specifically, in some courses female students are biased against while in some other courses male students are biased against in terms of at-risk detection. This observation is useful for future work on developing fair machine learning models in educational setting.

Overall, the main contributions of this work are summarized as

- We observe bias embedded in educational data and propose a fair machine learning model for identifying at-risk students based on metric-free individual fairness. The experimental results show the efficacy of the model at mitigating bias.
- From the experimental results, we find that different courses have different bias. This insight informs that future work on fairness should be built in a course specific manner. Trying to build a one-size-fit-all fair model might result in unfairness at course-level.

The rest of the paper is organized as following. Section 7.1 discusses related work on EDM and fairness. The following section introduce preliminary on the definition of individual fairness. In Section 7.3, we propose our fair models for at-risk students detection. Datasets and experimental protocol is described in Section 7.4. Section 7.5 presents experimental results and analysis. The last section concludes the paper and discusses future work.

7.1 Related Work

In this work, we focus on mitigating bias in classification tasks. We first describe related works in EDM that rely on classification. Then we describe the formalizations of fairness. Lastly, we talk about proposed methods for fair machine learning.

7.1.1 Classification Problems in EDM

In educational data mining, there are many tasks that can be formulated as a classification problem and many works have been proposed in this area such as affect detection [6], dropout prediction [99], graduation prediction [8], at-risk student detection [16, 82], knowledge tracing [19], etc.

Affect detection is the task of classifying a student's affective states such as boredom, confusion, delight, concentration and frustration by using sensor [7] and sensor-free [100] data. Vinayak et al. [101] proposed to predict student dropout using a Naive-Bayes classifier. Ojha et al. [9] proposed SVMs, Gaussian Processes and Deep Boltzmann Machines for student's graduation prediction using factors such as pre-university preparation. A set of human-interpretable features have been engineered by Polyzou et al. [16] for at-risk student detection. All these tasks can be formulated as a classification problem. However, all these works did not consider the potential bias and discrimination of the models. In this work, we try to build a general method that can be used for different kinds of tasks. To test the proposed method, we focus on the task of identifying at-risk student.

7.1.2 Fairness Formalizations

Over the years, different formalizations of fairness have been proposed that focus on different aspects. For example, statistical parity [94] requires that the probability of being predicted as positive across all the groups should be nearly the same. Equal odds imposes the constraint that the true positive rate should be the same for all the groups [95]. Equal opportunity requires a qualified individual should be predicted as qualified regardless of his/her sensitive attribute [95]. Another type of fairness formalization focuses more on individual level. The notion of individual fairness proposed by Cynthia et al. [97] assumes that similar individuals should be treated similarly. However, the requirement of a problem-specific similarity metric limits its adoption [102]. Hu et al. [98] proposed metric free individual fairness based on the assumption that the prediction outcome of an individual should be not be influenced by the individual's sensitive attribute. The elimination of similarity metric makes implementation of metric free individual fairness easier.

7.1.3 Fair Machine Learning Algorithms

Several algorithms have been proposed to achieve individual fairness. Based on John Rawls' notion of fair equality of opportunity, Joseph et al. [50] proposed an individual fairness notion that a worse individual should never be favored over a better one. To implement this notion, an algorithm based on contextual bandits have been proposed. The output from the contextual bandits is an interval indicating the qualification of an individual. Individuals with overlapping intervals are chained together and treated equally, while individuals with lower upper confidence bound are favored less. The unfairness comes from the prediction's dependence on sensitive attribute. To remove the dependence, Zemel et al. [51] proposed learning a fair representation which does not contain sensitive information. The representation is a cluster of embedding vectors. Following the idea of learning fair representation, Edwards [52] proposed to remove sensitive information from the learned representation by using adversarial learning. The input feature vectors are mapped to an embedding vector by an encoder. An adversary tries to predict the sensitive attribute from the representation. The encoder and the adversary plays a minimax game to remove sensitive information. The fair representation learning algorithms achieve individual fairness by first learn a representation and then train a classifier based on the learned representation. Our proposed model directly puts fairness constraints on the predictions.

7.2 Preliminaries

In this section, we discuss the formalization of individual fairness.

7.2.1 Individual Fairness

Cynthia et al. [97] introduces the concept of individual fairness, which is based on the idea that similar individuals should be treated similarly. This definition requires a similarity metric measuring the similarity between two individuals. Given two individuals x_i and x_j, a classifier H is individually fair if the difference of the predictions between the individuals are upper bounded by

their dissimilarity. The definition is as following

$$D(H(x_i), H(x_j)) < d(x_i, x_j) \tag{7.1}$$

where D is the distance measure between the outputs of the classifier and d is the distance metric between the two individuals. The drawback of this definition is that a similarity metric is required. A similarity metric guaranteeing fairness is problem specific and requires strong assumptions, which obstructs its adoption [102].

7.2.2 Metric Free Individual Fairness

Hu et al. [98] proposed metric free individual fairness based on the idea that the qualification of an individual should not be influenced by his/her sensitive attribute. Thus, changing an individual's sensitive attribute should not change the prediction of a classifier. The definition of metric free individual fairness is following

$$D(P(Y|x_i, S = s_i), P(Y|x_i, S \neq s_i)) < \epsilon \tag{7.2}$$

where s_i is the sensitive attribute of individual i, D is the distance measure of the predictions, ϵ is an arbitrarily small positive number. This definition eliminates the requirement of a similarity measure between individuals. In this work, we develop fair model based on this definition.

7.3 Methods

7.3.1 Problem Statement

In this work, we focus on the task of identifying at-risk students. Given a student i with $((x_i, s_i), y_i)$, $x_i \in \mathbb{R}^P$ encodes the student's grades in courses taken prior to the target course; $s_i \in \{0, 1\}$ is the student's sensitive attribute such as gender or race; $y_i \in \{0, 1\}$ is the ground truth label indicating whether a student is at-risk (1) or not (0). We focus on a binary sensitive attribute, though our method can be easily extend to scenarios where the sensitive attribute is n-ary. We want to build a

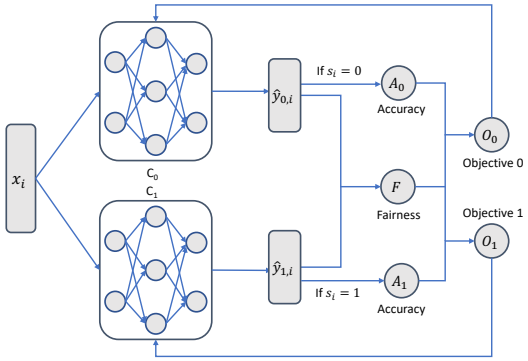


Figure 7.2: The architecture of the proposed model. The model consists of two classifiers C_0 and C_1 corresponding to sensitive attribute 0 and 1. An input vector x_i is fed into the two classifiers and the outputs are used to compute accuracy and fairness score. Note that if the sensitive attribute s_i is 0, accuracy A_0 and fairness F are combined to compute objective O_0 and only classifier C_0 is updated; otherwise, A_1 and fairness F are combined to form objective O_1 and classifier C_1 is updated.

classifier to predict if a student will underperform in a future target course. The classifier needs to satisfy two constraints: 1) make predictions as accurate as possible and 2) the output of the classifier is individually fair as specified by Equation 7.2.

The model is trained in a course-specific manner. Given a target course, we extract all the students who have taken it. The courses these students have taken prior to the target course are extracted as prior courses. The students' grades in the prior courses are extracted to form a matrix X and the students' grades in the target course are Y. Students' sensitive attributes are denoted as S. We train a course-specific model on $((X, S), Y)$ to predict whether students who have not taken the target course will fail it or not. Note that sensitive attributes S are not used as features.

7.3.2 Multiple Cooperative Classifier Model

In this section, we present the proposed model, multiple cooperative classifier model (MCCM). Figure 7.3 shows the architecture of the proposed model. The model is composed of two classifiers, each of which corresponds to a sensitive attribute, e.g., male or female. Given an individual $((x_i, s_i), y_i)$, the feature vector x_i is fed into the two classifiers. The output of the classifier corresponding to s_i is the individual's probability of being positive, while the output of the classifier

corresponding to $1 - s_i$ is the individual's probability of being positive if his/her sensitive attribute is changed. Based on the assumption of metric free individual fairness, to be fair the difference between the outputs of the two classifiers should be ignorable. In this work, the difference is the KL-divergence of the two outputs. In addition to fairness, we also care about the accuracy of the classifier. Therefore, for student i, the objective function we seek to optimize is as following

$$L_i = -y_i \log \hat{p}_{s_i,i} - (1 - y_i) \log(1 - \hat{p}_{s_i,i}) + \lambda \text{KL}(\hat{p}_{s_i,i}, \hat{p}_{1-s_i,i}) \tag{7.3}$$

where λ is a hyperparameter trading off between accuracy and fairness, $\hat{p}_{s_i,i}$ is the probability of being positive predicted by classifier s_i and $\hat{p}_{1-s_i,i}$ is the probability predicted by classifier $1 - s_i$. Note that, for L_i only the classifier corresponding to s_i is updated. The classifiers are feed-froward neural networks with two hidden layers. The activation function is chosen to be ReLU [103]. Dropout [104] is used to prevent overfitting.

Procedure 1 Multiple Cooperative Classifier Model

Input: Data $D = \{((x_i, s_i), y_i)\}_{i=1}^N$, learning rate α, λ, number of iterations T, classifier C_0 and C_1.
Initialize parameters $\{\theta_0^0, \theta_1^0\}$
1: **for** $t = 1, ..., T$ **do**
2: Sample example $((x_i, s_i), y_i)$ from D
3: Feed x_i into classifier C_{s_i} and C_{1-s_i}
4: Compute the loss L_i according to equation 7.3
5: $\theta_{s_i}^{t+1} = \theta_{s_i}^t + \alpha \frac{\partial L_i}{\partial \theta_{s_i}^t}$
6: **end for**
7: **return** $\{\theta_0^T, \theta_1^T\}$

7.3.3 Cooperative Contextual Bandits

Gradient Contextual Bandit

Contextual bandits algorithms make a decision at each step by estimating the action value $Q(x, a)$, which is the expected reward of taking action a given context x. In this section, we introduce gradient contextual bandits which learns a stochastic policy that maps a context vector to a probability distribution of actions. This is desirable in our case as we want to know the distance between two

distributions of actions given a context vector, which is detailed in Section 7.3.3. Besides that, a stochastic policy is beneficial in other domains such as recommender systems. For example, a user might like an action movie as much as a science fiction movie, which can be easily modeled by using a stochastic policy, while a deterministic policy can only choose the movie that the user likes the best.

Formally, we want to learn a policy $\pi_\theta(a|x) = P(A = a|X = x, \theta)$ parameterized by θ, which is the probability of choosing action a given context vector x that maximizes the expected reward

$$J(\theta) = E_{x\sim P(X)}[\pi_\theta(a|x)Q(x, a)] \tag{7.4}$$

In the following text, we ignore θ and use $\pi_\theta(a|x)$ and $\pi(a|x)$ interchangeably. As the action space is discrete, we can parameterize the policy as

$$\pi(a|x) = \frac{e^{\phi(x,a)}}{\sum_b e^{\phi(x,b)}} \tag{7.5}$$

where $\phi(x, a)$ is a preference score of choosing action a, which is mapped from feature vector x.

The gradient of the expected reward of a policy with respect to the policy parameter is

$$\nabla J(\theta) = E_{x\sim P(X)}[\nabla\pi_\theta(a|x)Q(x, a)] \tag{7.6}$$

However, the probability distribution of X is unknown. Instead, we seek to optimize the empirical reward $J_t(\theta)$, which is an unbiased estimator of the expected reward, namely, $J(\theta) = E[J_t(\theta)]$. The empirical policy gradient is

$$\nabla J_t(\theta) = \nabla \log(\pi_\theta(a_t|x_t)r_t(a_t)). \tag{7.7}$$

The gradient of the policy with respect to its parameters is the gradient of the logarithm of the policy

times the reward. The derivation of the policy gradient is detailed as following

$$\begin{aligned}
\nabla J_t(\theta) &= \sum_a \nabla \pi_\theta(a|x_t) q(x_t, a) \\
&= \sum_a \pi_\theta(a|x_t) \frac{\nabla \pi_\theta(a|x_t)}{\pi_\theta(a|x_t)} q(x_t, a) \\
&= E_{\pi_\theta}[\nabla \log \pi_\theta(a|x_t) q(x_t, a)] \\
&= E_{\pi_\theta}[\nabla \log \pi_\theta(a_t|x_t) q(x_t, a_t)] \\
&= E_{\pi_\theta}[\nabla \log \pi_\theta(a_t|x_t) R_t] \\
&= \nabla \log \pi_\theta(a_t|x_t) R_t.
\end{aligned} \tag{7.8}$$

where E_{π_θ} is the expectation taken with respect to π_θ.

The policy parameter can be updated by using stochastic gradient ascent algorithm as

$$\theta_{t+1} = \theta_t + \alpha \nabla \log \pi_\theta(a_t|x_t) r_t(a_t). \tag{7.9}$$

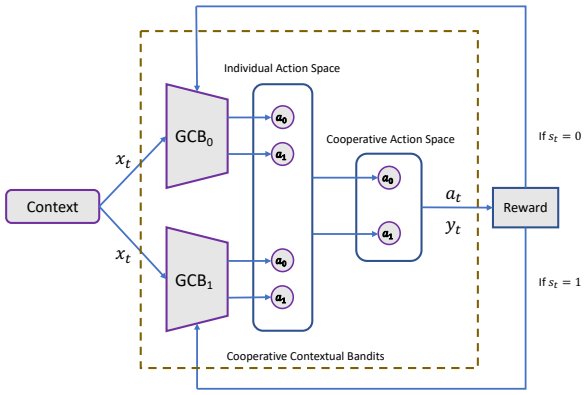


Figure 7.3: The Cooperative Contextual Bandits Model.

Metric free individual fairness imposes the constraint that changing the sensitive attribute of an

individual should not change the outcome. We propose cooperative contextual bandits algorithm which is composed of two gradient contextual bandits $B_s, s \in \{0, 1\}$, where s is the sensitive attribute. Figure 7.3 shows the model architecture of the proposed model. Each gradient contextual bandit corresponds to a sensitive attribute group, e.g., male or female. Each gradient contextual bandit learns a policy $\pi_s, s \in \{0, 1\}$ which maps a feature vector to an action distribution. Given an individual with context vector x_t, x_t is input into both bandits, each of which outputs an action distribution. The algorithm takes an action according to the action distribution of the bandit corresponding to s_t. A reward is determined by the ground truth label y_t and the distance between the action distributions of the two bandits. The bandit corresponding to s_t receives the reward feedback and gets updated. Note that the other bandit is not updated.

According to the definition of metric free individual fairness, the difference between outcomes given by different gradient contextual bandits should be ignored, i.e. $D(\pi_{s_i}(x_i), \pi_{1-s_i}(x_i)) < \epsilon$, for arbitrarily small ϵ. Since we are using gradient contextual bandits which directly maps the context vector to the probability distribution of actions, the difference of the outcomes can be measured by KL divergence between the outcomes.

Reward Function

At time step t, a individual (x_t, s_t) is presented, an action a_t is taken and a reward $r_t(a_t)$ is received. The action is the predicted outcome for the individual, i.e., whether he/she is hired. The reward function is composed of two parts, fairness and accuracy.

$$r_t(a_t) = r_t^a(a_t) - \lambda \text{KL}(\pi_{s_t}(x_t), \pi_{1-s_t}(x_t)) \tag{7.10}$$

where $r_t^a(a_t) \in \{0, 1\}$ is the accuracy reward, λ is a hyperparameter trading off between accuracy and fairness. The reward is high when the prediction is correct while the difference between the outcomes from different contextual bandits is small. KL divergence is a measure of how much a probability distribution is different from another distribution.

Model Architecture of Gradient Contextual Bandits

The input into the gradient contextual bandit is a feature vector describing an individual. The bandit learns a policy that maps the feature vector to an action distribution. The policy is parameterized as a feed-forward neural network. As we assume a binary discrete label, the last layer of the neural networks is a softmax layer whose output is the probability distribution of actions. The neural networks are composed of one hidden layer and the activation function is a ReLU function [105].

Algorithm 2 shows the detail of the proposed algorithm.

Procedure 2 Cooperative Contextual Bandits

Input: Data $D = \{((x_i, s_i), y_i)\}_{i=1}^{N}$, learning rate α, λ, number of steps T.
Initialize parameters $\{\theta_0^0, \theta_1^0\}$
1: **for** $t = 1, ..., T$ **do**
2: Sample example $((x_t, s_t), y_t)$ from D
3: Sample $a_t \sim B_{s_t}(\cdot|x_t)$
4: Take action a_t, compute reward $r_t(a_t)$ using Equation 7.10
5: $\theta_{s_t}^{t+1} = \theta_{s_t}^{t} + \alpha \nabla \log B_{s_t}(a_t|x_t) r_t(a_t)$
6: **end for**
7: **return** $\{\theta_0^T, \theta_1^T\}$

Convergence Analysis

In this section, we analyze the convergence property of the proposed algorithm.

Assumption 1. *There exists Q, such that the reward function satisfies*

$$|r_t(a_t)| \leq Q \tag{7.11}$$

for all t and a_t.

Assumption 1 states that the reward function is bounded. The reason is that $r_t^a(a_t) \in \{0, 1\}$ and $\mathrm{KL}(\pi_{s_t}(x_t), \pi_{1-s_t}(x_t))$ is bounded.

Assumption 2. *Let $s \in \{0, 1\}$. The first and second derivatives of the score function are element-wise bounded by constants F and S, respectively, and $0 \leq F, S < \infty$*

$$|\nabla \log \pi_s| \leq F \tag{7.12}$$

$$|\nabla^2 \log \pi_s| \leq S. \tag{7.13}$$

As the action space is discrete, the policy is parameterized as

$$\pi_s(a|x) = \frac{e^{\phi(x,a)}}{\sum_b e^{\phi(x,b)}}. \tag{7.14}$$

Therefore, $\nabla \log \pi_s = (1 - \pi_s(a|x))\nabla\phi(x, a)$, which is bounded if $\nabla\phi(x, a)$ and the parameter θ_s is bounded. The second derivative is derived as

$$\begin{aligned} \nabla^2 \log \pi_s(a|x) = (1 - \pi(a|x))^2 \nabla^2 \phi(x, a) \nabla^2 \pi_s(a|x) \\ - (1 - \pi(a|x))^2 \nabla\phi(x, a) \nabla\phi(x, a)^T \end{aligned} \tag{7.15}$$

which is also bounded if $\nabla^2\phi(x, a)$, $\nabla^2\pi_s(a|x)$ and $\nabla\phi(x, a)$, θ_s are bounded.

Lemma 1. *Under Assumption 2, for $\forall s \in \{0, 1\}$, $J_s(\theta)$ is L-Lipschitz smooth and $0 \leq L < \infty$*

$$||\nabla J_s(\theta_1) - \nabla J_s(\theta_2)|| \leq L||\theta_1 - \theta_2|| \tag{7.16}$$

Lemma 1 assures that the objective function $J_s(\theta)$ is smooth, which is essential in analyzing the convergence of the algorithm.

Proof The Hessian matrix of the score function $\log \pi_{\theta_s}$ is

$$\nabla^2 \log \pi_{\theta_s} = \pi_{\theta_s}^{-1} \nabla^2 \pi_{\theta_s} - \nabla \log \pi_{\theta_s} \nabla \log \pi_{\theta_s}^T. \tag{7.17}$$

Therefore,

$$\nabla^2 \pi_{\theta_s} = \pi_{\theta_s} \nabla^2 \log \pi_{\theta_s} + \pi_{\theta_s} \nabla \log \pi_{\theta_s} \nabla \log \pi_{\theta_s}^T. \tag{7.18}$$

The second derivative of the objective function is bounded as shown below

$$\begin{aligned} \nabla^2 J(\theta_s) &= \int_X \nabla^2 \pi_{\theta_s}(a|x) q(x,a) dx \\ &= \pi_{\theta_s} (\int_X \nabla^2 \log \pi_{\theta_s}(a|x) q(x,a) dx + \\ &\int_X \nabla \log \pi_{\theta_s}(a|x) \nabla \log \pi_{\theta_s}(x|a)^T q(x,a) dx) \\ &\leq SQ + F^2 Q = L. \end{aligned} \tag{7.19}$$

Bounded second derivative implies Lipschitz continuity, therefore $J(\theta_s)$ is L-smooth.

Theorem 1. *For $\forall s \in \{0,1\}$, let $\{\theta_s^t\}_{t=0}^T$ be the sequence of learned parameters of the policy π_s given by Algorithm 2. Then, under Assumption 1 and 2, we have*

$$||\nabla J(\theta_s^m)||_2^2 \leq \frac{1}{T} \sum_{t=0}^{T} ||\nabla J(\theta_s^k)||_2^2 \leq \frac{J(\theta_s^*) - J(\theta_s^0)}{MT}, \tag{7.20}$$

where $\nabla J(\theta_s^m) = \min_{t \in \{0,\dots,T\}} \nabla J(\theta_s^t)$, $J(\theta_s^*) = \max_{t \in \{0,\dots,T\}} J(\theta_s^t)$, and M is a constant with $0 < M < \infty$.

Theorem 1 shows that the average of the gradient norm square converges to a neighborhood near zero with rate of $\frac{1}{T}$ or the minimizer of the gradient norm square converges to zero in $O(\frac{1}{T})$ iterations. The minimizer can be obtained by using early stopping.

Proof The objective function $J(\theta_s)$ is smooth, which directly implies

$$\begin{aligned} J(\theta_s^{k+1}) &\geq J(\theta_s^k) + \langle \nabla J(\theta_s^k), \theta_s^{k+1} - \theta_s^k \rangle - \frac{L}{2}||\theta_s^{k+1} - \theta_s^k||_2^2 \\ &\geq J(\theta_s^k) + \alpha||\nabla J(\theta_s^k)||_2^2 - \frac{L}{2}||\theta_s^{k+1} - \theta_s^k||_2^2 \\ &\geq J(\theta_s^k) + \alpha||\nabla J(\theta_s^k)||_2^2 - \frac{L\alpha^2}{2}||\nabla J(\theta_s^k)||_2^2 \\ &\geq J(\theta_s^k) + \frac{2\alpha - L\alpha^2}{2}||\nabla J(\theta_s^k)||_2^2 \\ &\geq J(\theta_s^k) + M||\nabla J(\theta_s^k)||_2^2 \end{aligned} \tag{7.21}$$

where $M = \frac{2\alpha - L\alpha^2}{2}$.

By telescoping sum the above equation with k from 0 to K, we obtain

$$J(\theta_s^k) \geq J(\theta_s^1) + M\sum_{k=0}^{K}||\nabla J(\theta_s^k)||_2^2. \tag{7.22}$$

Therefore,

$$\begin{aligned} \frac{1}{K}\sum_{k=0}^{K}||\nabla J(\theta_s^k)||_2^2 &\leq \frac{J(\theta_s^k) - J(\theta_s^1)}{MK} \\ &\leq \frac{J(\theta_s^*) - J(\theta_s^1)}{MK} \end{aligned} \tag{7.23}$$

where $J(\theta_s^*)$ is the maximizer of $J(\theta_s^k)$ with respect to k.

Defining $\nabla J(\theta_s^m) = \min_{t\in\{0,\ldots,T\}} \nabla J(\theta_s^t)$, we obtain

$$||\nabla J(\theta_s^m)||_2^2 \leq \frac{1}{T}\sum_{t=0}^{T}||\nabla J(\theta_s^k)||_2^2 \leq \frac{J(\theta_s^*) - J(\theta_s^0)}{MT}. \tag{7.24}$$

This Completes the proof for Theorem 1

7.4 Experimental Protocol

7.4.1 Datasets

Table 7.1: Dataset Statistics

Major	#S	#C	#G	#M	#F	#AA	#NAA
BIOL	6,127	16	124,716	1,927(31.45%)	4,200(68.55%)	759(12.39%)	5,368(87.61%)
CEIE	450	7	23,708	338(75.11%)	112(24.89%)	27(6.00%)	423(94.00%)
CS	2,430	11	90,819	1,942(79.92%)	488(20.08%)	157(6.46%)	2,273(93.54%)
ECE	671	10	65,396	575(85.69%)	96(14.31%)	66(9.84%)	605(90.16%)
PSYC	5,110	17	84,504	1,200(23.48%)	3,910(76.52%)	694(13.58%)	4,416(86.42%)

#S total number of students, #C number of courses for prediction, #G total number of grades
#M number of male students, #F number of female students, #AA number of African-American students #NNA number of non-African-American students.

To evaluate the proposed model, we collect ten-year data at University X from Fall 2009 to Fall 2019. We choose top five majors including Biology (BIOL), Civil Engineering (CEIE), Computer Science (CS), Electrical Engineering (ECE) and Psychology (PSYC). We only choose a course if there are at least 300 students who have taken it. We use a student's grade in prior courses to predict whether a student is at-risk of failing a target course. While preprocessing the data, we exclude courses that are not relevant to a major such as elective courses of liberal arts. Table 7.1 shows statistics of the data. From the table, we can see clear gender difference for different majors. Female students tend to choose Biology and Psychology majors, while male students are more prone to engineering majors such as Civil Engineering, Computer Science and Electrical Engineering. Overall, the proportion of African-American students is relatively small, especially for Civil Engineering and Computer Science.

We build course specific models, namely, for a target course we train a classifier to predict whether a student will fail that course in the future. We define as at-risk student if the student's grade is lower than 3.0. Given a target course, the data related to that course is split into 75%, 15%, 15% for training, validation and testing, respectively.

7.4.2 Baselines

As in this work we focus on individual fairness, we compare our proposed model with several individually fair algorithms.

Logistic Regression (LR)

This baseline does not have fairness constraint. It directly predicts if a student is at-risk or not. The input is a feature vector encoding a student's grades in prior courses. The output is the student's probability of failing the target course.

Rawlsian Fairness (Rawlsian)

The concept of Rawlsian fairness is that a worse candidate should never be favored over a better one. Joseph et al. [50] proposed an individually fair algorithm utilizing contextual bandits as building block to implement Rawlsian fairness. Given an individual, the output from the contextual bandit is an interval indicating the qualification of the individual. Given a group of individuals, their intervals are chained together if they overlap. Chained individuals are treated identically and selected with equal probability. The hyperparameters for this model are regularization parameter λ selected from $\{1.0, 3.0, 9.0\}$ and the scaling parameter δ chosen from $\{0.3, 0.5, 0.9\}$.

Learning Fair Representation (LFR)

The unfairness of a prediction comes from the correlation of the output with the sensitive attribute. Zemel et al. [51] proposed to remove the correlation by learning an intermediate representation and train a classifier on it. The representation is composed of a cluster of prototypes. To remove sensitive information from the representation, a constraint is imposed that given two random individuals from the protected and advantaged group their probability of mapping to any prototype should be almost the same. The hyperparameters include A_x, A_y and A_z which are used to control a balance between the recovery of the original feature, accuracy and fairness. A_x is set to be 0.01, following the original work and A_y, A_z are chosen from $\{0.1, 0.5, 1, 5, 10\}$. The dimension of the representation is chosen from $\{10, 20, 30\}$.

Adversarial Learned Fair Representation (ALFR)

Edwards et al. [52] propose to remove sensitive information from representation by adversarial learning. A encoder maps the original feature vector to a latent embedding vector, from which an adversary tries to predict the sensitive attribute. While the adversary tries to predict the sensitive attribute, the encoder seeks to generate a representation that prevent the encoder from predicting it. The encoder and the adversary play a minimax game to remove sensitive information from the representation. The hyperparameters of this method include the dimension of the representation chosen from $\{10, 20, 30\}$. The encoder and the adversary are feed-froward neural networks consists of 2 hidden layers.

7.4.3 Evaluation Metrics

To evaluate if the proposed algorithm satisfy the accuracy and fairness constraints, we utilize three evaluation metrics **accuracy**, **discrimination** and **consistency**.

The **accuracy** metric assesses the predictive accuracy of the model, defined as following

$$\text{acc} = \frac{\sum_{i=1}^{N} \mathbb{1}(y_i = \hat{y}_i)}{N} \tag{7.25}$$

where N is the number of examples, $\hat{y}_i$ is the prediction and $\hat{y}$ is the ground truth label.

Discrimination measures the difference between the groups' rate of being predicted as positive, mathematically expressed as following

$$\text{discri} = |\frac{\sum_{i=1}^{N} \mathbb{1}(s_i = 0) * \hat{y}_i}{\sum_{i=1}^{N} \mathbb{1}(s_i = 0)} - \frac{\sum_{i=1}^{N} \mathbb{1}(s_i = 1) * \hat{y}_i}{\sum_{i=1}^{N} \mathbb{1}(s_i = 1)}| \tag{7.26}$$

Consistency compares the predicted results of an individual with his/her k-nearest neighbors. If the predicted results is close to the results of the neighbors, consistency is high and the algorithm

is fair. Consistency is defined as following

$$\text{consist} = 1 - \sum_{i=1}^{N} \frac{\sum_{n=1}^{K} |\hat{y}_i - \sum_{j \in \text{kNN}(x_i)} \hat{y}_j|}{K} \tag{7.27}$$

where **kNN**(x_i) is the k-nearest neighbors of individual i.

We use Gower similarity [106] to measure the similarity between individuals. Gower similarity is defined as

$$\text{Gower}(i, j) = \frac{\sum_{k=1}^{N} w_k S_{ijk}}{\sum_{k=1}^{N} w_k} \tag{7.28}$$

where N is the number of features and w_k is the weight of the k-th variable, in this paper the weights are set to one; S_{ijk} is the contribution by the k-th variable. If the k-th variable is continuous, S_{ijk} is defined as

$$S_{ijk} = 1 - \frac{|x_{ik} - x_{jk}|}{r_k} \tag{7.29}$$

where x_{ik} is the value of k-th feature of i and r_k is the range of values for the k-th variable. If the k-th variable is categorical, S_{ijk} is 1 if $x_{ik} = x_{jk}$ or 0, otherwise.

7.5 Experimental Results

7.5.1 Results and Analysis

We train a classifier for each course in a major to predict if a student will fail that course. The predictions are evaluated by using accuracy, discrimination and consistency. The results are averaged across the courses in a major. Table 7.2 shows the experimental results with gender as sensitive attribute. From the table, we can see that the proposed model **MCCM** achieves the best performance in mitigating bias in terms of discrimination. It is able to achieve both group fairness and individual fairness, although, it is designed for achieving individual fairness. The reason is that group and individual fairness are highly correlated so that achieving one helps achieving the other.

Table 7.2: Experimental results with gender as sensitive attribute.

Method	BIOL acc/discri/consist	CEIE acc/discri/consist	CS acc/discri/consist	ECE acc/discri/consist	PSYC acc/discri/consist
LR	0.7662/0.0613/0.8152	0.6761/0.0837/0.7451	0.6628/0.1007/0.7569	0.7545/0.0980/0.7655	0.7769/0.0192/0.9578
Rawlsian	0.5889/0.0807/0.8120	0.6250/0.0866/0.7052	0.5582/0.0913/0.8301	0.6660/0.1498/0.7036	0.7559/0.0960/0.9396
LFR	0.6470/0.0369/0.9691	0.6983/0.0518/0.9631	0.6004/0.0228/0.9463	0.7389/0.0273/0.9912	0.7898/0.0248/0.9865
ALFR	0.6802/0.0202/0.9675	0.7062/0.0240/0.9855	0.6124/0.0134/0.9821	0.7465/0.0114/0.9783	0.7903/0.0125/0.9878
MCCM	0.6774/0.0163/0.9401	0.6415/0.0165/0.9823	0.6180/0.0038/0.9562	0.7394/0.0061/0.9717	0.7868/0.0023/0.9958
CCB	0.6663/0.0089/0.9791	0.5901/0.0334/0.9624	0.6051/0.0257/0.9416	0.7300/0.0279/0.9790	0.7861/0.0094/0.9954

acc = accuracy, discri = discrimination, consist = consistency.

Table 7.3: Experimental results with race as sensitive attribute.

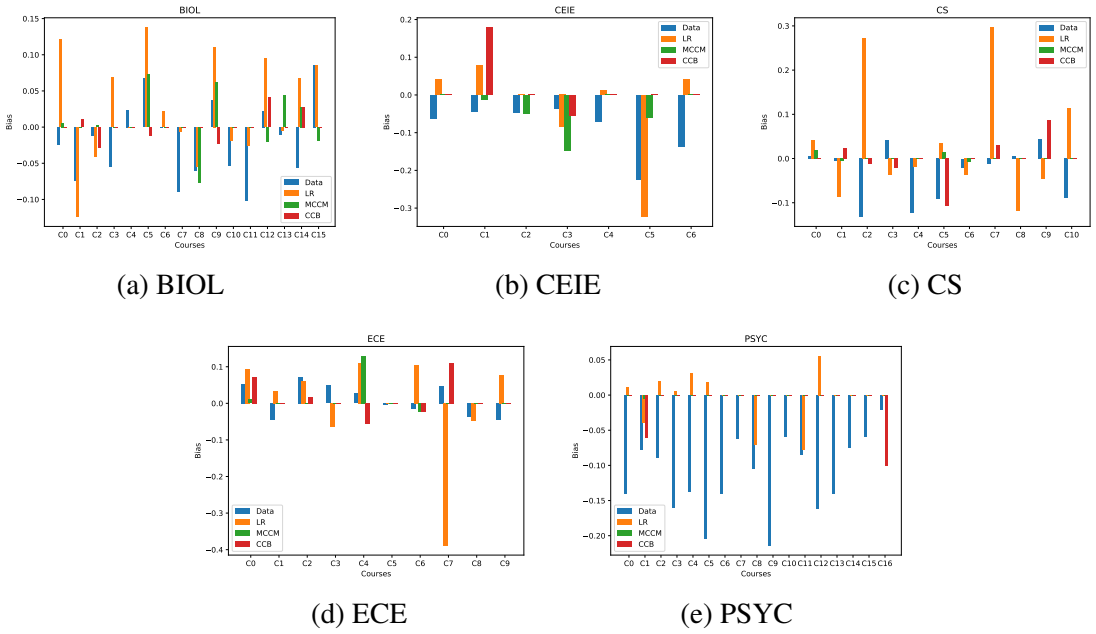
Method	BIOL acc/discri/consist	CEIE acc/discri/consist	CS acc/discri/consist	ECE acc/discri/consist	PSYC acc/discri/consist
LR	0.7662/0.1004/0.8152	0.6761/0.1411/0.7451	0.6628/0.1085/0.7569	0.7545/0.1238/0.7655	0.7769/0.0276/0.9578
Rawlsian	0.5854/0.1129/0.7870	0.5849/0.3658/0.7349	0.5561/0.1857/0.8007	0.6999/0.1446/0.7416	0.7608/0.0776/0.9570
LFR	0.6202/0.0569/0.9051	0.7099/0.1722/0.9701	0.6107/0.0599/0.9897	0.7441/0.0800/0.9852	0.7874/0.0172/0.9933
ALFR	0.6850/0.0505/0.9504	0.7274/0.0862/0.9688	0.6129/0.0086/0.9715	0.7435/0.0384/0.9887	0.7898/0.0156/0.9882
MCCM	0.6563/0.0198/0.9340	0.7138/0.0114/0.9828	0.5895/0.0303/0.9968	0.7133/0.0013/0.9986	0.7857/0.0021/0.9974
CCB	0.6443/0.0115/0.9767	0.6781/0.1071/0.8798	0.5944/0.0372/0.9824	0.7440/0.0137/0.9842	0.7863/0.0042/0.9993

acc = accuracy, discri = discrimination, consist = consistency.

The predictions from **LR** model is highly biased as there is no fairness constraint imposed on it, but it performs well with respect to predicting accuracy. On average, the discrimination of **LR** is 7.3%. Other methods achieve fairness at the cost of accuracy. It is interesting to see that **Rawlsian** is not able to remove bias and in some cases it leads to even more unfair predictions. **Rawlsian** is based on the idea that a worse candidate should never be favored over a better one, which is implemented by interval chaining that is a weak fairness constraint. We can also observe from the table that different majors have different level of bias, e.g., Psychology has the least bias while Computer Science has the highest bias with respect to the predictions of **LR**.

The experimental results with race as sensitive attribute is shown in Table 7.3. The results are similar to those with gender as sensitive attribute. However, the bias exhibited is higher than that of treating gender as sensitive attribute. The predictions from bias-mitigating algorithms also exhibit higher bias than their predictions for using gender as sensitive attribute. The fair algorithms are able to remove bias to some extent. The proposed model **MCCM** achieves the best performance in terms of removing discrimination at the expense of predicting accuracy.

7.5.2 Fine-grained analysis of the bias



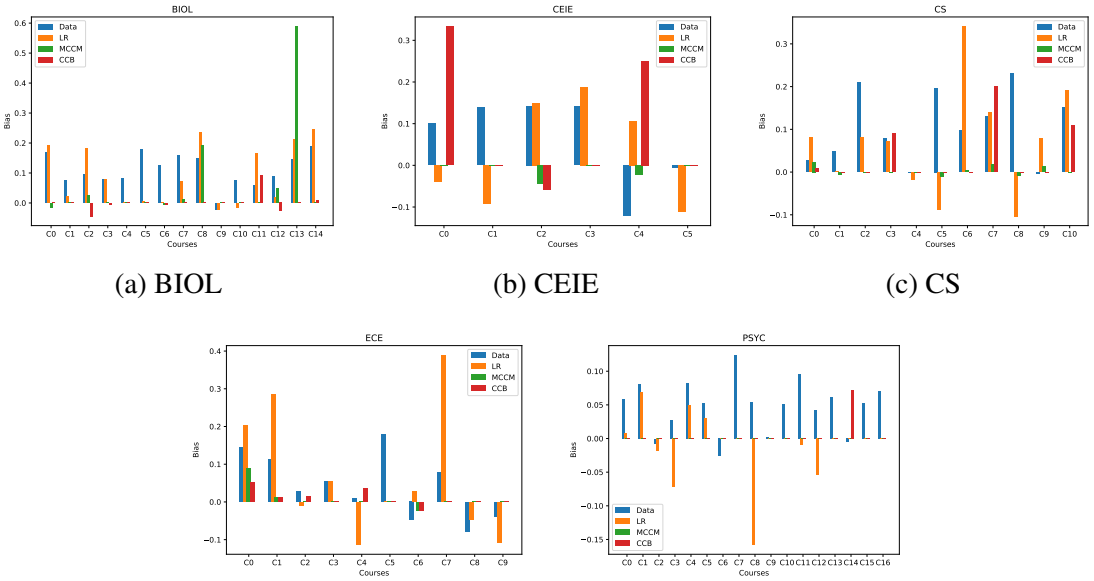
(a) BIOL (b) CEIE (c) CS

(d) ECE (e) PSYC

Figure 7.4: Bias of different courses with gender as sensitive attribute.

To have a fine-grained view of the bias, we look at the data and predictions at the course level. In this section, we analyze the bias embedded in the data and predictions from **LR** and the proposed model **MCCM**. Figure 7.4 and 7.5 shows the fine-grained results with gender and race as sensitive attribute, respectively. For Figure 7.4, the data bias is the proportion of at-risk female students subtracts the proportion of at-risk male students. Positive bias means female students are more likely to be predicted as at-risk; otherwise male students are more likely to be predicted as at-risk. For the predictions from the models, the bias is the female students' average probability of being predicted as at-risk students subtract that of male students. For Figure 7.5, the bias has similar definition, except that the sensitive attribute is race.

First of all, the overall data such as overall GPA by gender and race shows that male and African-American group are minority groups. However, when looking at the course level, different courses



(a) BIOL (b) CEIE (c) CS

(d) ECE (e) PSYC

Figure 7.5: Bias of different courses with race as sensitive attribute.

have different minority groups. Figure 7.4 and 7.5 show that in some courses male and African-American students are less likely to be at-risk. This insights can be used to inform future fairness work in educational data mining that a course specific model is desirable, considering that different courses have different minority groups. From the figures, we can also observe that data and machine learning models might have different bias direction. For example, in Figure 7.4(a), for course C0 the data bias is against male while **LR** and **MCCM** is against female. In addition, data bias does not necessarily leads to predictive bias. For example in Figure 7.4, all the courses show data bias. However, a no-fairness-constraint classifier, e.g., logistic regression has fair predictions in many courses.

7.5.3 Residual Bias

In this work, we build separate models to remove bias with respect to gender or race. We want to investigate how a model is biased with respect to gender when trained using race as sensitive attribute or vice versa. We define this kind of bias as residual bias. We evaluate the proposed model

Table 7.4: Predicting results with respect to race using model trained with gender as sensitive attribute.

Method	BIOL acc/discri/consist	CEIE acc/discri/consist	CS acc/discri/consist	ECE acc/discri/consist	PSYC acc/discri/consist
LFR	0.5438/0.3289/0.7089	0.5282/0.3743/0.7916	0.5067/0.3813/0.8694	0.5592/0.2873/0.8422	0.5479/0.2334/0.6460
ALFR	0.4912/0.0346/0.9212	0.4075/0.0198/0.8913	0.5474/0.0249/0.9528	0.4888/0.0841/0.9003	0.5006/0.0527/0.9462
MCCM	0.7058/0.0672/0.8912	0.7003/0.0856/0.9113	0.6315/0.0216/0.9491	0.7383/0.0376/0.9477	0.7884/0.0063/0.9942
CCB	0.6616/0.0346/0.9839	0.6009/0.0380/0.9495	0.6086/0.0333/0.9438	0.7277/0.0806/0.9700	0.7816/0.0171/0.9815

acc = accuracy, discri = discrimination, consist = consistency.

Table 7.5: Predicting results with resepct to gender using model trained with race as sensitive attribute.

Method	BIOL acc/discri/consist	CEIE acc/discri/consist	CS acc/discri/consist	ECE acc/discri/consist	PSYC acc/discri/consist
LFR	0.5359/0.3158/0.7708	0.4981/0.3521/0.7348	0.5504/0.2094/0.8175	0.5992/0.4244/0.8537	0.5440/0.1604/0.7678
ALFR	0.5667/0.0112/0.9588	0.4867/0.0182/0.8585	0.4679/0.0216/0.9166	0.6057/0.0177/0.9684	0.4965/0.0093/0.9613
MCCM	0.6885/0.0811/0.8838	0.5991/0.1254/0.9297	0.6309/0.0673/0.8515	0.7118/0.0537/0.9493	0.7880/0.0211/0.9787
CCB	0.6348/0.0421/0.9580	0.6356/0.1936/0.8379	0.5891/0.0507/0.9714	0.7457/0.0430/0.9748	0.7841/0.0108/0.9854

acc = accuracy, discri = discrimination, consist = consistency.

along with **LFR** and **ALFR**. In Table 7.4, the models are first trained to remove bias with respect to race and evaluate their bias by treating gender as sensitive attribute. Table 7.5 shows the results of models trained to remove bias with respect to gender and evaluated with respect to race. From the tables, we can see that **LFR** has the highest residual bias, while **ALFR** and **MCCM** has much lower residual bias. For psychology major, the residual bias is trivial for **MCCM** in Table 7.4 and for both in Table 7.5. However, in many cases there are still non-trivial residual bias for the models. A naive idea to mitigate residual bias is to treat gender and race as sensitive attribute simultaneously. However, the drawback of this approach is that there may be not enough training samples for some combinations, which leads to highly imbalanced data. Previous studies show that imbalanced data can result in biased classification errors for minority groups [15]. We leave the work of removing residual bias to future work.

7.6 Conclusion and Future Work

The concerns about bias and discrimination of machine learning models are rising with the increasing of their adoption. In educational setting, we observe bias from a real-world dataset and machine learning models without fairness constraints exhibit non-ignorable biased predictions. Machine learning models are intended to aid students with their learning. However, unfair treatment of students can undermine their learning and graduation. To mitigate discrimination in educational data mining, in this paper, we proposed a fair machine learning model satisfying metric-free individual fairness. We evaluate the model's performance on removing unfairness on datasets collected from an anonymous University. The results show the efficacy of the model on removing bias. Compared to other domains, educational data mining has its own characteristics. For example, in our dataset, when looking at university level, male and African-American students are biased against. However, at course level, different courses have different bias direction. This insights inform that future work on fairness in educational data mining should design course-specific models. In this work, we treat gender and race separately and the experimental results show that there is residual bias. In the future, we want to build models that treat gender and race as sensitive attributes simultaneously.

Bibliography

Bibliography

[1] R. Stillwell and J. Sable, "Public school graduates and dropouts from the common core of data: School year 2009-10. first look (provisional data). nces 2013-309." *National Center for Education Statistics*, 2013.

[2] N. Anschuetz. (2015) Breaking the 4-year myth: Why students are taking longer to graduate? [Online]. Available: http://college.usatoday.com/2015/12/16/breaking-the-4-year-myth-why-students-are-taking-longer-to-graduate/

[3] A. Elbadrawy, A. Polyzou, Z. Ren, M. Sweeney, G. Karypis, and H. Rangwala, "Predicting student performance using personalized analytics," *Computer*, vol. 49, no. 4, pp. 61–69, 2016.

[4] M. Sweeney, H. Rangwala, J. Lester, and A. Johri, "Next-term student performance prediction: A recommender systems approach," *arXiv preprint arXiv:1604.01840*, 2016.

[5] A. Polyzou and G. Karypis, "Grade prediction with course and student specific models," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2016, pp. 89–101.

[6] T.-Y. Yang, R. S. Baker, C. Studer, N. Heffernan, and A. S. Lan, "Active learning for student affect detection," in *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*. ERIC, 2019, pp. 208–217.

[7] L. Paquette, J. Rowe, R. Baker, B. Mott, J. Lester, J. DeFalco, K. Brawner, R. Sottilare, and V. Georgoulas, "Sensor-free or sensor-full: A comparison of data modalities in multi-channel affect detection." *International Educational Data Mining Society*, 2016.

[8] S. Hutt, M. Gardener, D. Kamentz, A. L. Duckworth, and S. K. D'Mello, "Prospectively predicting 4-year college graduation from student applications," in *Proceedings of the 8th international conference on learning analytics and knowledge*, 2018, pp. 280–289.

[9] T. Ojha, G. L. Heileman, M. Martinez-Ramon, and A. Slim, "Prediction of graduation delay based on student performance," in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 3454–3460.

[10] A. Benjamin, "Factors influencing learning," 2014.

[11] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, 2016, pp. 1050–1059.

[12] J. Angwin, J. Larson, S. Mattu, and L. Kirchner, "Machine bias," *ProPublica*.

[13] Z. Obermeyer, B. Powers, C. Vogeli, and S. Mullainathan, "Dissecting racial bias in an algorithm used to manage the health of populations," *Science*, vol. 366, no. 6464, pp. 447–453, 2019.

[14] M. Ali, P. Sapiezynski, M. Bogen, A. Korolova, A. Mislove, and A. Rieke, "Discrimination through optimization: How facebook's ad delivery can lead to skewed outcomes," *arXiv preprint arXiv:1904.02095*, 2019.

[15] J. Buolamwini and T. Gebru, "Gender shades: Intersectional accuracy disparities in commercial gender classification," in *Conference on fairness, accountability and transparency*, 2018, pp. 77–91.

[16] A. Polyzou and G. Karypis, "Feature extraction for classifying students based on their academic performance." *International Educational Data Mining Society*, 2018.

[17] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk, "Sparse factor analysis for learning and content analytics," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1959–2008, 2014.

[18] R. Agrawal, M. Christoforaki, S. Gollapudi, A. Kannan, K. Kenthapadi, and A. Swaminathan, "Mining videos from the web for electronic textbooks," in *International Conference on Formal Concept Analysis*. Springer, 2014, pp. 219–234.

[19] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon, "Individualized bayesian knowledge tracing models," in *International Conference on Artificial Intelligence in Education*. Springer, 2013, pp. 171–180.

[20] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User modeling and user-adapted interaction*, vol. 4, no. 4, pp. 253–278, 1994.

[21] A. Elbadrawy and G. Karypis, "Domain-aware grade prediction and top-n course recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 2016, pp. 183–190.

[22] H. Bydžovská, "Student performance prediction using collaborative filtering methods," in *International Conference on Artificial Intelligence in Education*. Springer, 2015, pp. 550–553.

[23] S. E. Sorour, K. Goda, and T. Mine, "Student performance estimation based on topic models considering a range of lessons," in *International Conference on Artificial Intelligence in Education*. Springer, 2015, pp. 790–793.

[24] M. A. Al-Barrak and M. Al-Razgan, "Predicting students final gpa using decision trees: a case study," *International Journal of Information and Education Technology*, vol. 6, no. 7, p. 528, 2016.

[25] S. Umair and M. M. Sharif, "Predicting students grades using artificial neural networks and support vector machine," in *Encyclopedia of Information Science and Technology, Fourth Edition*. IGI Global, 2018, pp. 5169–5182.

[26] H. Bydžovská, "Are collaborative filtering methods suitable for student performance prediction?" in *Portuguese Conference on Artificial Intelligence*. Springer, 2015, pp. 425–430.

[27] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, 2009.

[28] A. Polyzou and G. Karypis, "Grade prediction with models specific to students and courses," *International Journal of Data Science and Analytics*, vol. 2, no. 3-4, pp. 159–171, 2016.

[29] Z. Ren, X. Ning, and H. Rangwala, "Grade prediction with temporal course-wise influence," *arXiv preprint arXiv:1709.05433*, 2017.

[30] S. Morsy and G. Karypis, "Cumulative knowledge-based regression models for next-term grade prediction," in *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 2017, pp. 552–560.

[31] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains," *The annals of mathematical statistics*, vol. 41, no. 1, pp. 164–171, 1970.

[32] L. E. Baum and J. A. Eagon, "An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology," *Bulletin of the American Mathematical Society*, vol. 73, no. 3, pp. 360–363, 1967.

[33] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[34] G. Balakrishnan and D. Coetzee, "Predicting student retention in massive open online courses using hidden markov models," *Electrical Engineering and Computer Sciences University of California at Berkeley*, 2013.

[35] C. Geigle and C. Zhai, "Modeling student behavior with two-layer hidden markov models," *JEDM-Journal of Educational Data Mining*, vol. 9, no. 1, pp. 1–24, 2017.

[36] J. D. Ferguson, "Variable duration models for speech," *Proceedings of the Symposium on the Applications of Hidden Markov Models to Text and Speech*, pp. 143–179, 1980.

[37] J. Bulla, "Application of hidden markov models and hidden semi-markov models to financial time series," 2006.

[38] I. E. Livieris, K. Drakopoulou, and P. Pintelas, "Predicting students' performance using artificial neural networks," in *8th PanHellenic Conference with International Participation Information and Communication Technologies in Education*, 2012, pp. 321–328.

[39] T. Gedeon and S. Turner, "Explaining student grades predicted by a neural network," in *Neural Networks, 1993. IJCNN'93-Nagoya. Proceedings of 1993 International Joint Conference on*, vol. 1. IEEE, 1993, pp. 609–612.

[40] T.-Y. Yang, C. G. Brinton, C. Joe-Wong, and M. Chiang, "Behavior-based grade prediction for moocs via time series neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 5, pp. 716–728, 2017.

[41] F. OKUBO, T. YAMASHITA, A. SHIMADA, and S. KONOMI, "Students' performance prediction using data of multiple courses by recurrent neural network."

[42] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein, "Deep knowledge tracing," in *Advances in Neural Information Processing Systems*, 2015, pp. 505–513.

[43] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[44] Y. Gal and Z. Ghahramani, "On modern deep learning and variational inference," in *Advances in Approximate Bayesian Inference workshop, NIPS*, vol. 2, 2015.

[45] S. Barocas and A. D. Selbst, "Big data's disparate impact," *Calif. L. Rev.*, vol. 104, p. 671, 2016.

[46] S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq, "Algorithmic decision making and the cost of fairness," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 797–806.

[47] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," in *Advances in neural information processing systems*, 2016, pp. 3315–3323.

[48] M. J. Kusner, J. Loftus, C. Russell, and R. Silva, "Counterfactual fairness," in *Advances in Neural Information Processing Systems*, 2017, pp. 4066–4076.

[49] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, "Fairness through awareness," in *Proceedings of the 3rd innovations in theoretical computer science conference*. ACM, 2012, pp. 214–226.

[50] M. Joseph, M. Kearns, J. Morgenstern, S. Neel, and A. Roth, "Rawlsian fairness for machine learning," *arXiv preprint arXiv:1610.09559*, vol. 1, no. 2, 2016.

[51] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork, "Learning fair representations," in *International Conference on Machine Learning*, 2013, pp. 325–333.

[52] H. Edwards and A. Storkey, "Censoring representations with an adversary," *arXiv preprint arXiv:1511.05897*, 2015.

[53] M. Sweeney, J. Lester, and H. Rangwala, "Next-term student grade prediction," in *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE, 2015, pp. 970–975.

[54] GMU, "George mason university catalog," 2017. [Online]. Available: http://catalog.gmu.edu/

[55] UMN, "University of minnesota twin cities undergraduate catalog," 2017. [Online]. Available: http://www.catalogs.umn.edu/ug/index.html

[56] M. Schrepp, "About the connection between knowledge structures and latent class models," *Methodology*, vol. 1, no. 3, pp. 93–103, 2005.

[57] T. V. Duong, H. H. Bui, D. Q. Phung, and S. Venkatesh, "Activity recognition and abnormality detection with the switching hidden semi-markov model," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 838–845.

[58] Y. Guédon, "Estimating hidden semi-markov chains from discrete sequences," *Journal of Computational and Graphical Statistics*, vol. 12, no. 3, pp. 604–639, 2003.

[59] Q. Hu and H. Rangwala, "Enriching course-specific regression models with content features for grade prediction," 2017.

[60] S. Mangal, *Advanced educational psychology*. PHI Learning Pvt. Ltd., 2002.

[61] F. Rosenblatt, "Principles of neurodynamics. perceptrons and the theory of brain mechanisms," CORNELL AERONAUTICAL LAB INC BUFFALO NY, Tech. Rep., 1961.

[62] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[63] V. I. Arnold, "On functions of three variables," in *Dokl. Akad. Nauk SSSR*, vol. 114, no. 4, 1957, pp. 679–681.

[64] A. N. Kolmogorov, "On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition," *Translations American Mathematical Society*, vol. 2, no. 28, pp. 55–59, 1963.

[65] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 1135–1144.

[66] J. Li, W. Monroe, and D. Jurafsky, "Understanding neural networks through representation erasure," *arXiv preprint arXiv:1612.08220*, 2016.

[67] A. Avati, K. Jung, S. Harman, L. Downing, A. Ng, and N. H. Shah, "Improving palliative care with deep learning," *arXiv preprint arXiv:1711.06402*, 2017.

[68] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[69] "Undergraduate Retention and Graduation Rates," https://nces.ed.gov/programs/coe/indicator_ctr.asp.

[70] R. S. Baker and K. Yacef, "The state of educational data mining in 2009: A review and future visions," *JEDM— Journal of Educational Data Mining*, vol. 1, no. 1, pp. 3–17, 2009.

[71] C. Romero, S. Ventura, M. Pechenizkiy, and R. S. Baker, *Handbook of educational data mining*. CRC press, 2010.

[72] C. Lang, G. Siemens, A. Wise, and D. Gasevic, *Handbook of learning analytics*. SOLAR, Society for Learning Analytics and Research, 2017.

[73] S. Morsy and G. Karypis, "A study on curriculum planning and its relationship with graduation gpa and time to degree," in *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*. ACM, 2019, pp. 26–35.

[74] B.-H. Kim, E. Vizitei, and V. Ganapathi, "Gritnet: Student performance prediction with deep learning," *arXiv preprint arXiv:1804.07405*, 2018.

[75] Q. Hu and H. Rangwala, "Reliable deep grade prediction with uncertainty estimation," *arXiv:1902.10213*, 2019.

[76] B. Bakhshinategh, O. R. Zaiane, S. ElAtia, and D. Ipperciel, "Educational data mining applications and tasks: A survey of the last 10 years," *Education and Information Technologies*, vol. 23, no. 1, pp. 537–553, 2018.

[77] A. M. Shahiri, W. Husain *et al.*, "A review on predicting student's performance using data mining techniques," *Procedia Computer Science*, vol. 72, pp. 414–422, 2015.

[78] A. Elbadrawy, R. S. Studham, and G. Karypis, "Collaborative multi-regression models for predicting students' performance in course activities," in *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*. ACM, 2015, pp. 103–107.

[79] R. Wang, G. Harari, P. Hao, X. Zhou, and A. T. Campbell, "Smartgpa: how smartphones can assess and predict academic performance of college students," in *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing*. ACM, 2015, pp. 295–306.

[80] G. Balakrishnan, "Predicting student retention in massive open online courses using hidden markov models," Master's thesis, EECS Department, University of California, Berkeley, May 2013. [Online]. Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2013/EECS-2013-109.html

[81] V. Swamy, A. Guo, S. Lau, W. Wu, M. Wu, Z. Pardos, and D. Culler, "Deep knowledge tracing for free-form student code progression," in *International Conference on Artificial Intelligence in Education*. Springer, 2018, pp. 348–352.

[82] Q. Hu and H. Rangwala, "Course-specific markovian models for grade prediction," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2018, pp. 29–41.

[83] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *arXiv preprint arXiv:1801.07829*, 2018.

[84] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[85] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017.

[86] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," *arXiv preprint arXiv:1707.01926*, 2017.

[87] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[88] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[89] C. Raffel and D. P. Ellis, "Feed-forward networks with attention can solve some long-term memory problems," *arXiv preprint arXiv:1512.08756*, 2015.

[90] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[91] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[92] P. Sapiezynski, V. Kassarnig, and C. Wilson, "Academic performance prediction in a gender-imbalanced environment," 2017.

[93] S. Doroudi and E. Brunskill, "Fairer but not fair enough on the equitability of knowledge tracing," in *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, 2019, pp. 335–339.

[94] M. Feldman, S. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, "Certifying and removing disparate impact," 2014.

[95] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," *CoRR*, vol. abs/1610.02413, 2016. [Online]. Available: http://arxiv.org/abs/1610.02413

[96] M. J. Kusner, J. Loftus, C. Russell, and R. Silva, "Counterfactual fairness," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 4066–4076. [Online]. Available: http://papers.nips.cc/paper/6995-counterfactual-fairness.pdf

[97] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. S. Zemel, "Fairness through awareness," *CoRR*, vol. abs/1104.3913, 2011. [Online]. Available: http://arxiv.org/abs/1104.3913

[98] Q. Hu and H. Rangwala, "Cooperative contextual bandits for metric-free individual fairness," 2020.

[99] Y. Chen, A. Johri, and H. Rangwala, "Running out of stem: a comparative study across stem majors of college students at-risk of dropping out early," in *Proceedings of the 8th international conference on learning analytics and knowledge*, 2018, pp. 270–279.

[100] A. F. Botelho, R. S. Baker, and N. T. Heffernan, "Improving sensor-free affect detection using deep learning," in *International Conference on Artificial Intelligence in Education*. Springer, 2017, pp. 40–51.

[101] V. Hegde and P. Prageeth, "Higher education student dropout prediction and analysis through educational data mining," in *2018 2nd International Conference on Inventive Systems and Control (ICISC)*. IEEE, 2018, pp. 694–699.

[102] A. Chouldechova and A. Roth, "The frontiers of fairness in machine learning," *CoRR*, vol. abs/1810.08810, 2018. [Online]. Available: http://arxiv.org/abs/1810.08810

[103] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 315–323.

[104] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[105] R. H. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit," *Nature*, vol. 405, no. 6789, pp. 947–951, 2000.

[106] J. C. Gower, "A general coefficient of similarity and some of its properties," *Biometrics*, pp. 857–871, 1971.

Curriculum Vitae

Qian Hu received his Bachelor of Engineering degree in Information Engineering from Nanjing University of Aeronautics and Astronautics in 2010. He was a Graduate Teaching Assistant at Computer Science Department at George Mason University from August 2015 to May 2016. After May 2016, he has been working under the supervision of Dr. Huzefa Rangwala as a Graduate Research Assistant at Data Mining Laboratory at Computer Science Department at George Mason University.