Data-driven Methods for Course Selection and Sequencing

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Sara Morsy

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor Of Philosophy

Prof. George Karypis

May, 2019

Acknowledgements

This thesis was done with the support and help of so many wonderful people, whom I would like to thank from the bottom of my heart.

First of all, I would like to thank my advisor George Karypis. He is an amazing teacher and researcher, and he really shows his interest in teaching all of his lab members on a very personal level. I have learned a lot from him, from research skills, to data mining and machine learning, to scientific writing and critical thinking. He has spent a lot of effort with me going deep into the small details of every piece of the research to teach me how to do everything right. He has been a great mentor to me, and has supported me through all the challenging times I have faced during my PhD. I could not thank him enough for all what he did with me.

I would like to thank Professors Rui Kuang, Vipin Kumar, Nikolaos Sidiropoulos, Zizhuo Wang, and Lana Yarosh for serving on my thesis and preliminary exams. They have such deep knowledge in the field that I was fortunate enough to have them on my committees. Their comments and feedback have been tremenedously useful to shaping my research.

A very special thanks goes to both my parents, who have supported me to a great extent during my PhD, and without whom I would have faced many more challenges during that period. Both of them have flied a very long distance just to come and support me when I gave birth to my daughter, until she grew old enough to go to a daycare. My husband, Hesham, has always been a great supporter for me and had taught me a lot of programming skills that helped me during my PhD. Being both PhD students during most of my PhD was such a positive thing for both of us. A special thanks to my little daughter, Jana, who has been the main source of joy and inspiration for me the moment she was born. I would also like to thank my siblings, Salma and

Moamen, who have been always supporting me and giving me positive comments when I needed them the most.

The PhD journey could have been a lot harder without my second Egyptian family whom I knew when I first went to Minnesota and we started our PhD journey together. Specifically, I would like to thank Amr, Eman, and Reem for everything and every moment we spent together. This second family also grew bigger when Ibrahim, Heba, Ahmed, and Sara joined us in Minnesota later. Our times together, in the bus shuttle going to and from campus, at one of our homes, and at the many places and restaurants we went to will always be great memories for me. Also, giving birth to our first kids at similar times and watching them grow and play together and spending most of their days together at the daycare will always be invaluable to me.

I was also lucky to belong to a wonderful research lab consisting of many bright minded and supportive people, where we spent good times together helping each other in research and in coding, as well as enjoying our free time together. Specifically, I would like to thank: Agi, Ancy, Asmaa, Eva, Maria, Shalini, Xia, David, Dominique, Haoji, Jeremy, Mohit, Rezwan, Santosh, Saurav, and Shaden. I would also like to thank a special friend from another lab: Cheng Jin, for all the funny times we spent together and all her help in Minnesota and in California as well.

Last, a great thanks to the staff the Department of Computer Science, the Digital Technology Center, and the Minnesota Supercomputing Institute at the University of Minnesota for providing the resources which were crucial for my research and for helping me so many times on so many things.

Dedication

To my parents, Faiza and Ahmed. To my love, Hesham. To my sweet little daughter, Jana. To my brother, Moamen. To my sister, Salma.

Abstract

Learning analytics in higher education is an emerging research field that combines data mining, machine learning, statistics, and education on learning-related data, in order to develop methods that can improve the learning environment for learners and allow educators and administrators to be more effective. The vast amount of data available about students' interactions and their performance in classrooms has motivated researchers to analyze this data in order to gain insights about the learning environment for the ultimate goal of improving undergraduate education and student retention rates. In this thesis, we focus on the problem of course selection and sequencing, where we would like to help students make informed decisions about which courses to register for in their following terms. By analyzing the historical enrollment and grades data, this thesis studies the two main problems of course selection and sequencing, namely grade prediction and course recommendation. In addition, it analyzes the relationship between degree planning in terms of course timing and ordering and the students' GPA and time to degree.

First, we focus on predicting the grades that students will obtain on future courses so that they can make informed decisions about which courses to register for in their following terms. We model the grade prediction problem as cumulative knowledge-based linear regression models that learn the courses' required and provided knowledge components and use them to estimate a student's knowledge state at each term and predict the grades that he/she can obtain on future courses.

Second, we focus on improving the knowledge-based regression models we previously developed by modeling the complex interactions among prior courses using non-linear and neural attentive models, in order to have more accurate estimation of a student's knowledge state. In addition, we model the interactions between a target course, which we would like to predict its grade, and the other courses taken concurrently with it. We hypothesize that concurrently-taken courses can affect a student's performance in a target course, and thus modeling their interactions with that course should lead to better predictions.

Third, we focus on analyzing the degree plans of students to gain more insights about how course timing and sequencing relate to their GPAs and time to degree. Toward this end, we define several course timing and course sequencing metrics and compare different sub-groups of students who have achieved high vs low GPA as well as sub-groups of students who have graduated on time vs over time.

Fourth, we focus on improving course recommendation by recommending to each student a set of courses which he/she is prepared for and expected to perform well in. We model this problem as a grade-aware course recommendation problem, where we propose two different approaches. The first approach ranks the courses by using an objective function that differentiates between courses that are expected to increase or decrease a student's GPA. The second approach combines the grades predicted by grade prediction methods with the rankings produced by course recommendation methods to improve the final course rankings. To obtain the course rankings in both approaches, we adapted two widely-used representation learning techniques to learn the optimal temporal ordering between courses.

In summary, this thesis addresses two closely related problems by: (1) developing cumulative knowledge-based regression models for grade prediction; (2) developing context-aware non-linear and neural attentive knowledge-based models for grade prediction; (3) analyzing degree planning and how the time when students take courses and how they sequence them relate to their GPAs and time to degree; and (4) developing novel approaches for grade-aware course recommendation.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The average six-year graduation rate across four-year higher-education institutions has been around 59% over the past 15 years [1, 2], while less than half of college graduates finish within four years [2]. These statistics pose challenges in terms of workforce development, economic activity and national productivity. This has resulted in a critical need for analyzing the available data about past students in order to provide actionable insights to improve college student graduation and retention rates.

Learning analytics (LA) is an emerging research field that spans the areas of data mining, machine learning, statistics, and education in order to analyze educational-related data and help understand the dynamics of such data. The goal of LA is to improve teaching and learning by generating patterns to characterize learner's habits, predicting his/her responses and providing timely feedback, which is done by developing statistical and machine learning methods that learn from the historical raw data [3].

This thesis focuses on LA in higher education institutions to help undergraduate students and their advisors during the process of course selection and sequencing. Towards these goals, this thesis addresses the problems of grade prediction and course recommendation. First, we develop linear regression models that can predict the grades for future courses. Second, we develop context-aware non-linear and neural attentive models that improve upon the linear regression grade prediction models that we developed in the past. Third, we analyze the degree plans taken by students and study how their course timing and ordering relate to their GPAs and time to degree. Fourth, we propose a grade-aware course recommendation framework that recommends to students courses

that will help them towards finishing their degree requirements in a timely fashion and maintaining or improving their overall GPAs.

1.1 Key Contributions

There are two main problems associated with course selection and sequencing in undergraduate education. The first is the grade prediction problem, which aims to predict the student's grade in a course that he/she is interested in taking. The second is the course recommendation problem, which aims to recommend to each student a set of courses that align with his/her degree requirements. In recent years, grade prediction and course recommendation problems have gained a lot of interest due to the increasing demand to analyze the available data about past students and help improve the students' graduation and retention rates. Therefore, development of accurate grade prediction and course recommendation methods is highly desired. In addition, analysis of degree planning helps us in deriving deep insights about how course timing and ordering relate to the students' GPAs and time to degree.

1.1.1 Cumulative Knowledge-based Regression Models (CKRM)

Many academic programs offer flexible degree plans, that include a small number of required core courses and a large number of elective courses. These electives allow students to customize their degree plans to better match their career goals. Existing methods suffer from their ability to perform well in such flexible degree programs.

In this thesis (Chapter 5), we present a new set of Cumulative Knowledge-based Regression Models (CKRM), that mainly builds on the following idea. Each degree program requires a set of courses that need to be taken in some suggested sequence such that the knowledge provided by the earlier courses are essential for students to be able to perform well in more advanced courses. Towards this end, CKRM assumes that there is a space of knowledge components describing the overall curriculum. Within that space, each course is modeled via a *knowledge component vector* that contains the knowledge components that it provides. A knowledge component can be provided by a single or multiple courses. A student by taking a course acquires its knowledge components in a way that depends on the grade that he/she obtains in that course.

CKRM models the knowledge that a student has acquired after taking a set of courses via a *knowledge state vector* that is computed as the sum of the knowledge component vectors of these courses weighted by the grades that he/she has obtained in them. In order to predict the grade that a student will obtain on a specific course, CKRM estimates a per-course linear model that captures the knowledge components that are required in order to perform well in that course. Given the student's knowledge state vector prior to taking a course and that course's estimated linear model, the predicted grade is obtained as the dot-product of these two vectors.

There are two main contributions from the CKRM-based methods. First, it models the way an academic degree program is designed in a natural way such that the knowledge offered from previously-taken courses collectively contribute to the student's predicted grade in future courses. Second, it is able to identify the knowledge required from students to perform well in different courses, which can help in course sequencing as well as assist students by providing them with information about the required knowledge for performing well in courses.

1.1.2 Context-aware Non-linear and Neural Attentive Knowledge-based Models

Though the CKRM method that we developed in Chapter 5 was shown to provide state-of-the-art grade prediction accuracy, it is limited in that it learns shallow linear models that may not be able to accurately capture the complex interactions among prior courses. In addition, it does not consider the effect of the concurrently-taken courses on a student's performance in a target course.

In this thesis (Chapter 6), we propose context-aware non-linear and neural attentive knowledge-based models, which improve upon the CKRM models that we previously developed (Chapter 5) from two perspectives: (i) using non-linear and neural attentive models to better estimate the student's knowledge state; and (ii) modeling the interactions between a target course and the other courses taken concurrently with it. For estimating the student's knowledge state, we explore two different approaches. First, we develop a non-linear model, MAximum Knowledge-based model (MAK), where we hypothesize that each course provides knowledge at a certain knowledge level. MAK estimates a student's knowledge state by employing a maximum-based pooling layer

along each component of the prior courses' embeddings. Second, we develop a Neural Attentive Knowledge-based model, NAK, where we hypothesize that prior courses should have different contribution towards a target course. The attention weights are computed using two different activation functions. The first, called the softmax activation function, is the most commonly-used function, which converts a given input vector of real weights to a probability distribution. The second, called the sparsemax activation function, was recently proposed to truncate the smaller weighted values to zero, hence producing sparse attention weights. This is useful when the input contains some relevant and some irrelevant objects to the object of interest. For modeling the interactions between a target and concurrent courses, we hypothesize that the knowledge provided by concurrent courses modify the knowledge required by a target course. We aggregate the concurrent course embeddings using non-linear and neural attentive models and then estimate a context-aware embedding for the target course.

A comprehensive set of results show that: (i) the proposed context-aware non-linear and neural attentive models outperform other baseline methods, including the previously-developed CKRM method, with statistically significant improvements; (ii) the context-aware non-linear model outperforms the context-aware neural attentive model and all baselines in making less severe under-predictions; (iii) estimating a student's knowledge state via a non-linear or neural attentive model significantly outperforms estimating it via a linear model; (iv) learning sparse attention weights for the neural attentive model outperforms learning soft weights; (v) modeling the interactions between a target course and concurrent courses significantly improve the performance of the non-linear model and gives similar performance for the neural attentive model; and (vi) the neural attentive model was able to uncover the listed and hidden pre-requisite courses for target courses.

1.1.3 Analysis of How Course Timing and Sequencing Relate to Students' GPAs and Time to Degree

Student success in undergraduate education is mainly measured by his/her graduation GPA and time to degree. Several course recommendation methods have been developed to help students in selecting courses that align with their degree requirements. These methods use all the past students' data to train their models, regardless of the students'

GPA or time to degree. Other studies have investigated the effect of many variables on the time to degree. These variables include: family background, prior academic achievement, working status (on- or off-campus), ... etc. None of these studies have studied the effect of degree planning, i.e., when a student takes his/her courses and how he/she sequences them, on the time to degree.

In this thesis (Chapter 7), we study the relationship between degree planning, in term of course timing and ordering, and each of the student's GPA and time to degree. We define several metrics to measure course timing and similarity in course sequencing between pairs of students. We then measure these metrics for different GPA- and time-to-degree-based groups of students, and compare their values among these different groups.

Our analysis on a large-scale real-world dataset show that: (i) low time to degree students tend to take more courses ahead of time, and follow more similar sequencing for the common courses (especially in their later years), than high TTD students; and (ii) low GPA students tend to take more courses ahead of time, and follow more diverse sequencing for the common courses, than high GPA students.

In addition, we propose new course timing and ordering features to use in time to degree prediction. We train several binary classification models using the proposed course timing and ordering features and show that degree planning is a good indicator for TTD prediction.

1.1.4 Grade-aware Course Recommendation Approaches

Both course recommendation and grade prediction methods aim to help students during the process of course registration in each semester. By learning from historical registration data, course recommendation focuses on recommending courses to students that will help them in completing their degrees. Grade prediction focuses on estimating the students' expected grades in future courses. Based on what courses they previously took and how well they performed in them, the predicted grades give an estimation of how well students are prepared for future courses. Nearly all of the previous studies have focused on solving each problem separately, though both problems are inter-related in the sense that they both aim to help students graduate in a timely and successful manner.

In this thesis (Chapter 8), we propose a new grade-aware course recommendation framework that focuses on recommending a set of courses that will help students: (i) complete their degrees in a timely fashion, and (ii) maintain or improve their GPA. To this end, we propose two different approaches for recommendation. The first approach ranks the courses by using an objective function that differentiates between courses that are expected to increase or decrease a student's GPA. The second approach uses the grades that students are expected to obtain in future courses to improve the ranking of the courses produced by course recommendation methods. The proposed framework combines the benefits of both course recommendation and grade prediction approaches to better help students graduate in a timely and successful manner.

To obtain course rankings in the first approach, we adapt two widely-known representation learning techniques, which have proven successful in many fields, to solve the grade-aware course recommendation problem. The first is based on Singular Value Decomposition (SVD), which is a linear model that learns a low-rank approximation of a given matrix. The second, which we refer to as Course2vec, uses a log-linear model to formulate the problem as a maximum likelihood estimation problem. In both approaches, the courses taken by each student are treated as temporally-ordered sets of courses, and each approach is trained to learn these orderings.

A comprehensive set of results show that: (i) the proposed grade-aware course recommendation approaches outperform grade-unaware course recommendation methods in recommending more courses that increase the students' GPA and fewer courses that decrease it; and (ii) the proposed representation learning approaches outperform competing approaches for grade-aware course recommendation in terms of recommending courses which students are expected to perform well in, as well as differentiating between courses which students are expected to perform well in and those which they are expected not to perform well in.

1.2 Outline

This thesis is organized as follows:

- Chapter 2 provides definitions and notations that are used throughout this thesis.

- Chapter 3 presents the background and existing methods related to the grade prediction and course recommendation problems.
- Chapter 4 discusses the metrics used for evaluating the grade prediction methods, which are proposed in Chapter 5 and Chapter 6.
- Chapter 5 presents a new set of Cumulative Knowledge-based Regression Models (CKRM) for solving the grade prediction problem.
- Chapter 6 presents the proposed context-aware non-linear and neural attentive knowledge-based models for grade prediction.
- Chapter 7 presents a large-scale analysis on degree planning and how course timing and sequencing relate the students' GPA and time to degree.
- Chapter 8 presents the proposed grade-aware course recommendation approaches to recommend to students courses that align with their degree requirements and that help them maintain or improve their GPAs.
- Chapter 9 summarizes the main contributions of this thesis and outlines some future research directions.

1.3 Related Publications

The work presented in this thesis and the related work has been published in leading conferences and journals in the fields of data mining and information retrieval. The related publications are listed as follows:

- **Sara Morsy** and George Karypis. Accounting for Language Changes over Time in Document Similarity Search. In *ACM Transactions on Information Systems* (ACM TOIS), pages 1–26, 2016.
- **Sara Morsy** and George Karypis. Cumulative Knowledge-based Regression Models for Next-term Grade Prediction. In *Proceedings of SIAM International Conference on Data Mining* (SDM), 2017.

- **Sara Morsy** and George 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 and Knowledge* (LAK), 2019.

- **Sara Morsy** and George Karypis. Will This Course Increase or Decrease Your GPA? Towards Grade-aware Course Recommendation. In *Journal of Educational Data Mining* (JEDM), 2019 (accepted for publication).

- **Sara Morsy** and George Karypis. Neural Attentive Knowledge-based Models for Grade Prediction. In *Proceedings of the 12th International Conference on Educational Data Mining* (EDM), 2019 (accepted for publication).

- **Sara Morsy** and George Karypis. Context-aware Non-linear and Neural Attentive Knowledge-based Models for Grade Prediction. under submission.

Chapter 2

Notations and Definitions

Boldface uppercase letters will be used to represent matrices (e.g., $\mathbf{G}, \mathbf{R}$) and boldface lowercase letters to represent row vectors, (e.g., $\mathbf{r}$). The ith row of matrix $\mathbf{R}$ is represented as $\mathbf{r}_i$. The entry in the ith row and jth column of matrix $\mathbf{G}$ is denoted as $g_{i,j}$. A predicted value is denoted by having a hat over it (e.g., $\hat{g}$).

$\mathcal{S}$ and $\mathcal{C}$ are used to denote the sets of students and courses, respectively, whose respective cardinalities are m and n (i.e., $|\mathcal{S}| = m$ and $|\mathcal{C}| = n$). Matrix $\mathbf{G}$ will represent the $m \times n$ student-course grades matrix, where $g_{s,c}$ denotes the grade that student s obtained in course c. A student s enrolls in sets of courses in consecutive terms, numbered relative to s from 1 to the number of terms in he/she has enrolled in the dataset. A set $\mathcal{T}_{s,w}$ will denote the set of courses taken by student s in term w.

Chapter 3

Background and Related Work

With the alarming reported statistics on undergraduate graduation and retention rates, where around 59% of first-time, full-time undergraduate students at four-year institutions graduate within six years, and 19.5% drop out from these institutions [1], there has been a critical need to improve these graduation and retention rates. LA in higher education aims to analyze the historical raw data that is available about past students to understand the underlying factors for their success/failure in order to assist current and future students graduate in a timely and successful fashion.

Researchers have been applying different techniques to solve several related problems in LA. These problems include (but are not limited to): predicting the student's performance and detecting his/her behavior [4, 5], identifying at-risk students [6, 7], analyzing the contents of discussion forums [8, 9], predicting the grades for course activities [10], knowledge tracing and student modeling [11–13], clustering similar students based on their learning preferences and interactions patterns [14, 15], and others.

In the following sections, we discuss some of the state-of-the-art grade prediction and course recommendation methods. In addition, we review other research areas that are relevant to our work in this thesis.

3.1 Grade Prediction

Regression Methods Polyzou *et al.* [4] proposed two regression-based methods: Course-Specific and Student-Specific Regression models (namely; CSR (or CSR) and

SSR, respectively). CSR is based on the fact that the student's performance in a future course is based on his performance in the past courses. Consider a student s that has taken j courses $\langle c_1, \ldots, c_j \rangle$ in that sequence, and a course c that s has not yet taken for which we will like to predict his/her grade. In CSR, the grade for student s in course c is predicted as a sparse linear combination of his previous grades, which is computed as

$$\hat{g}_{s,c} = b_c + \mathbf{r}_c \Big(\sum_{i=1}^{j} g_{s,c_i} \mathbf{z}_{c_i} \Big)^T,$$

where b_c is a course bias term, $\mathbf{r}$ and $\mathbf{z}$ are vectors of dimension equal to the total number of courses n, $\mathbf{r}_c$ is a linear model associated with course c, g_{s,c_i} is the grade that student s obtained on course c_i, and $\mathbf{z}_{c_i}$ is an indicator vector with one in the dimension corresponding to course c_i. Since CSR treats each course as having a unique dimension that does not share anything with any other course, it assumes that each course provides a set of knowledge components that are totally different from any other course, which does not hold for many courses. The capability of CSR to accurately model the accumulation of knowledge decreases as the flexibility of the degree program increases, i.e., as students can take more diverse courses that provide the same or similar knowledge components prior to taking the target course.

SSR [4] tries to overcome this limitation by estimating course-specific linear regression models that are also specific to each student. These student-specific models are learned by only using similar students who have taken a sufficient number of common courses for the target student. However, as their results showed, the performance of SSR is highly dependent on the percentage of common courses between previous and target students (overlap ratio) and is thus limited to target students with a high overlap ratio.

Sweeney *et al.* [5] used different regression-based methods, namely; Random Forest (RF), Stochastic Gradient Descent Regression, k-Nearest Neighbor (kNN) and Personalized Multi-Linear Regression (PMLR), on a set of extracted features about students, courses and instructors. They found that both RF and PMLR perform well compared to kNN.

Matrix Factorization (MF) Methods Low rank Matrix Factorization methods have been successful in predicting ratings in the context of recommender systems. Similar to the user-item rating matrix, grade prediction can be modeled via MF by constructing a student-course grade matrix and learning low rank representations for both students and courses. This low rank representation can be thought of as representing the knowledge space for both students and courses. Thus, the grade that student s can obtain on course c can be estimated as

$$\hat{g}_{s,c} = \mu + sb_s + cb_c + \mathbf{u}\ \mathbf{v}^T, \tag{3.1}$$

where μ, sb_s and cb_c are the global, student and course bias terms, respectively, and $\mathbf{u}$ and $\mathbf{v}$ are the student and course latent vectors, respectively. The parameters of the MF model () are estimated by using the squared loss function with L2 regularization:

$$\underset{\mu,\mathbf{sb},\mathbf{cb},\mathbf{U},\mathbf{V}}{\text{minimize}}\ \frac{1}{2}\sum_{s,c\in\mathbf{G}}(g_{s,c}-\hat{g}_{s,c})^2 + \frac{\alpha}{2}\Big(\ \|\mathbf{sb}\|_2^2 + \|\mathbf{cb}\|_2^2 + \|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2\ \Big),$$

where: $\mathbf{sb}$ and $\mathbf{cb}$ are the student and course bias vectors, respectively, and $\mathbf{U}$ and $\mathbf{V}$ are the student and course latent factor matrices, respectively.

Since the accurate recovery of the low rank model assumes that the observed entries are drawn randomly from the matrix, and since this assumption does not hold for the student-course grade data (since there is a clear structure for students and courses where students can select only a few subset of courses based on their majors and academic levels), Polyzou *et al.* [4] also proposed a course-specific matrix factorization (CSMF) method. CSMF estimates an MF model for each course by utilizing a course-specific subset of the whole student-course grade matrix. Specifically, for a target course c and a set of students $\mathcal{S}^c$ for which we need to estimate their grade for c, it uses the students who took c and their grades prior to taking c as well as $\mathcal{S}^c$ (their grades prior to taking c) to build the student-course grade matrix for MF. However, both MF and CSMF performed poorly when compared to CSR [4]. We believe this is due to the inaccurate estimation of the student latent representation, since students took a few number of courses, especially those who are still in their freshman or sophomore years.

Elbadrawy *et al.* [16] defined various academic features for both students and courses

on different granularity levels and incorporated them in MF methods. Examples of this grouping are: academic level and major for students, and course level and subject for courses. These groups were used to estimate different bias terms for different MF models for the corresponding student and course groups, where in each model, the student and course group bias terms were estimated by replacing the individual student and course bias terms by them in Eq. 3.1. The different MF models were then combined to estimate the final predicted grade.

3.2 Course Recommendation

Different machine learning methods have been recently developed for course recommendation. For example, [17] used association rule mining to discover significant rules that associate academic courses from previous students' data. [18] ranked the courses for each student based on the course's importance within his/her major, its satisified prerequisites, and the extent by which the course adds to the student's knowledge state.

Another set of recommendation methods proposed in [19–22] focused on satisfying the degree plan's requirements that include various complex constraints. The problem was shown to be NP-hard and different heuristic approaches were proposed in order to solve the problem.

Elbadrawy *et al.* [16] proposed using both student- and course-based academic features, in order to improve the performance of three popular recommendation methods in the education domain, namely: popularity-based ranking, user-based collaborative filtering and matrix factorization. These features are used to define finer groups of students and courses and were shown to improve the recommendation performance of the three aforementioned methods than using coarser groups of students.

The group popularity ranking method proposed in [16] and referred to as **grp-pop**, ranks the courses based on how frequently they were taken by students of the same major and academic level as the target student. Though this is a simple ranking method, it was shown to be among the best performing methods proposed by the authors. This is due to the domain restrictions, where each degree program offers a specific set of required and elective courses for the students to choose a subset from, and a pre-requisite structure exists among most of these courses.

Pardos *et al.* [23] proposed a course2vec model that used a skip-gram neural network architecture. Their model takes as input one course, and outputs multiple probability distributions over the courses.

Backenkohler *et al.* [24] proposed to combine grade prediction with course recommendation. They used a course dependency graph constructed using the Mann-Whitney U-test as the course recommendation method. This graph consists of nodes that represent courses and directed edges between them. A directed edge going from course A to course B means that the chance of getting a better grade in B is higher when A is taken before B than when A is not taken before B. One limitation of this approach is that, for pairs (A, B) of courses that do not have sufficient data about A not being taken before B, no directed edge will exist from A to B, despite the fact that there may be sufficient data about A followed by B, which may imply that A is a pre-requisite for B.

3.3 Representation Learning

Representation learning has been an invaluable approach in machine learning and artificial intelligence for learning from different types of data such as text and graphs. Objects can be represented in a vector space via local or distributed representations. Under local (or one-hot) representations, each object is represented by a binary vector, of size equal to the total number of objects, where only one of the values in the vector is one and all the others are set to zero. Under distributed representations, each object is represented by a dense or sparse vector, which can come from hand-engineered features that is usually sparse and high-dimensional, or a learned representation, called "embeddings" in a latent space that preserves the relationships between the objects, which is usually low-dimensional and more practical than the former.

A widely used approach for learning object embeddings is Singular Value Decomposition (SVD) [25]. SVD is a traditional low-rank approximation method that has been used in many fields. In recommendation systems, a user-item rating matrix is typically decomposed into the user and item latent factors that uncover the observed ratings in the matrix, e.g., [26–29].

Recently, neural networks have gained a lot of interest for learning object embeddings in different fields, for their ability to handle more complex relationships than SVD. Some

of the early well-known architectures include Word2vec [30] and Glove [31], which were proposed for learning distributed representations for words [30]. For instance, neural language models for words, phrases and documents in Natural Language Processing, e.g., [30–34] are now widely used for different tasks, such as machine translation and sentiment analysis. Similarly, learning embeddings for graphs, such as: DeepWalk [35], LINE [36] and node2vec [37] were shown to have performed well on different applications, such as: multi-label classification and link prediction. Moreover, learning embeddings for products in e-commerce and music playlists in cloud-based music services have been recently proposed for next basket recommendation [38–40].

3.3.1 Neural Attentive Models

Neural networks have been used extensively in many fields, including, but not limited to: Natural Language Processing [41, 42] and recommender systems [43–46]. The attention mechanism has been recently introduced to neural network modeling and was shown to improve the performance of different models. Instead of aggregating the input object embeddings via a summation or mean pooling function, which assumes equal contribution of all objects, the idea is to allow the selected objects to contribute differently when compressing them to a single representation. Neural attentive networks have been successfully applied in many recommendation system techniques, such as factorization machines [43, 44], item-based collaborative filtering [46], and user-based collaborative filtering [47].

Part of our work in Chapter 6 relies on the attention mechanism, and leverages several advances in this area. The most commonly-used activation function for the attention mechanism is the softmax function, which is easily differentiable and gives soft posterior probabilities that normalize to 1. A major disadvantage of the softmax function is that it assumes that each object contributes to the compressed representation, which may not always hold in some domains. To solve this, we need to output sparse posterior probabilities and assign zero to the irrelevant objects. Martins *et al.* [48] proposed the sparsemax activation function, which has the benefit of assigning zero probabilities to some output variables that may not be relevant for making a decision. This is done by defining a threshold, below which small probability values are truncated to zero. We also leverage the controllable sparsemax activation function recently

proposed by Laha *et al.* [49] that controls the desired degree of sparsity in the output probabilities. This is done by adding an L2 regularization term that is to be maximized in the loss function. This will potentially encourage larger probability values for some objects, moving the rest to zero.

Chapter 4

Evaluation Metrics for Grade Prediction

The grading system used by the University of Minnesota uses a 12 letter grade system (i.e., A, A-, B+, ... F). We will refer to the difference between two successive letter grades (e.g., B+ vs B) as a *tick*. We converted the predicted grades into their closest letter grades. We assessed the performance of the different approaches based on the Root Mean Squared Error (RMSE) as well as how many ticks away the predicted grade is from the actual grade, which is referred to as "Percentage of Tick Accuracy", or PTA. We computed the percentage of grades predicted with no error (zero tick), within one tick, and within two ticks, which will be referred to as PTA0, PTA1, and PTA2, respectively.

In general, the grades that are predicted with at most one or two ticks error are sufficiently accurate for the task of course selection whereas the grades that are predicted with an error of three or more ticks can incorrectly influence course selection.

Chapter 5

CKRM: Cumulative Knowledge-based Regression Models for Grade Prediction

5.1 Introduction and Motivation

A natural way to model the problem of grade prediction is to model the way the academic degree programs are structured. Each degree program requires a set of courses that need to be taken in some suggested sequence such that the knowledge provided by the earlier courses are essential for students to be able to perform well in more advanced courses. As we explained in Sec. 3.1, Polyzou *et al.* [4] proposed a Course-Specific Regression Model (CSR) which builds on this idea. However, CSR's underlying model cannot correctly capture the students' state of knowledge when the same knowledge can be acquired by taking different subsets of courses. As a result, its prediction performance deteriorates for programs with flexible degree plans.

In this chapter, we develop Cumulative Knowledge-based Regression Models (CKRM) that also builds on the idea of accumulating knowledge but addresses the aforementioned limitation of CSR. CKRM assumes that there is a space of knowledge components describing the overall curriculum. Within that space, each course is modeled via a *knowledge component vector* that contains the knowledge components that it provides. A

knowledge component can be provided by a single or multiple courses. A student by taking a course acquires its knowledge components in a way that depends on the grade that he/she obtains in that course. CKRM models the knowledge that a student has acquired after taking a set of courses via a *knowledge state vector* that is computed as the sum of the knowledge component vectors of these courses weighted by the grades that he/she has obtained in them. In order to predict the grade that a student will obtain on a specific course, CKRM estimates a per-course linear model that captures the knowledge components that are required in order to perform well in that course. Given the student's knowledge state vector prior to taking a course and that course's estimated linear model, the predicted grade is obtained as the dot-product of these two vectors.

We investigated three different ways of constructing the knowledge component space. Two of them construct the knowledge space in terms of an automatically identified latent space and the third uses the free text descriptions of the courses to extract keywords that form the space's dimensions. The difference between the two latent spaces is that one imposes the constraint that courses from different departments do not share any knowledge components, whereas the other one does not.

In the following sections, we present the summary of our results in Sec. 5.2. Then, we explain our methods in Sec. 5.3, describe the experimental setup and evaluation methodology in Section Sec. 5.4, discuss the results in Sec. 5.5 and summarize the Chapter in Sec. 5.6.

5.2 Main Contributions

Our contributions in this chapter are three-fold.

1. We propose a cumulative knowledge-based method for the problem of next-term grade prediction that better models the structure of degree programs and is better suited for flexible degree programs.

2. We performed an extensive experimental evaluation on a real world dataset containing 14 years worth of student grades from 12 academic departments from the College of Science and Engineering at University of Minnesota. This evaluation

showed that the proposed methods perform statistically significantly better than competing approaches.

3. We showed that the models that were estimated based on the extracted keywords can identify the knowledge that is required in order to perform well in a course, which is not captured by the course pre-requisites. This can be used to inform changes in course sequencing and degree programs.

5.3 Proposed Models

Consider a student s that has taken j courses $\langle c_1, \ldots, c_j \rangle$ in that sequence, and a course c that s has not yet taken for which we will like to predict his/her grade. A course c is assumed to provide a set of knowledge components that the student acquires after taking c. These knowledge components can be the set of topics or concepts taught by the course. We assume that all courses can be represented in a knowledge space of these different components. We will refer to the knowledge component vector of a course c as its *provided knowledge component vector* and we will denote it as $\mathbf{p}_c$. We define the *knowledge state* for student s after taking j courses as the knowledge he/she has acquired so far in the different knowledge components provided by the j courses. A student's s knowledge state after taking j courses will be denoted by the *knowledge state vector* $\mathbf{k}_{s,j}$ and will be computed as

$$\mathbf{k}_{s,j} = \sum_{i=1}^{j} \Big(\xi(s, c_j, c_i)\ g_{s,c_i}\ \mathbf{p}_{c_i} \Big), \tag{5.1}$$

where g_{s,c_i} is the grade that student s obtained on course c_i, and $\xi(s, c_j, c_i)$ is a time-based exponential decaying function designed to de-emphasize courses that were taken a long time ago. Equation 5.1 models a student's knowledge state as the sum of the provided knowledge component vectors of the courses he/she has taken so far, weighted by his/her grades in them. The grade-based weighting is designed to capture the fact that a student better acquires the knowledge components of a course on which he/she obtained a good grade than a course on which he/she did not.

The decaying function that we used is:

$$\xi(s, c_j, c_i) = e^{-\lambda(t_{s,c_j} - t_{s,c_i})}, \tag{5.2}$$

where λ is a user-specified non-negative parameter that controls the shape of the exponential decaying function, and t_{s,c_i} is the term number when student s took course c_i. This term number is encoded as follows. For each student, we encode his/her first term as the term numbered as 1, and each following term number is incremented by 1. This technique applies a time-based decaying weight on the prior courses, and is designed to model the fact that students tend to forget the knowledge components that they have acquired in courses that were taken a long time ago.

CKRM computes the grade that student s will obtain on a course c by applying a course-specific linear model $\mathbf{r}_c$ on the student's knowledge state vector prior to taking c. That is, the predicted grade is given by

$$\hat{g}_{s,c} = b_c + \mathbf{r}_c \, \mathbf{k}_{s,j}^T, \tag{5.3}$$

where b_c is a course bias term and $\mathbf{k}_{s,j}$ is the student's knowledge state vector. These course-specific linear models are estimated from the historical grade data and can be considered as capturing and weighting the knowledge components that a student needs to have accumulated in order to perform well in a course. For this reason, we will refer to these linear models as the *required knowledge component vectors*.

5.3.1 The Course Knowledge Component Space

In order to capture the knowledge components provided by courses, we investigated three different ways of defining the structure of the knowledge component space. Two of them are based on a latent space, and the third one is based on the textual descriptions of these courses.

Latent Knowledge Component Space

The most straightforward way to define the latent knowledge component space is to use the standard latent structure in which all dimensions, i.e., knowledge components,

are shared across all courses. We will refer to the CKRM-based method that uses the standard latent structure as **CKRMall**. For academic courses that belong to different departments, however, they should not share their provided knowledge components among each other. For instance, a course that belongs to Mechanical Engineering in general should not share any of its provided knowledge components with a course from Computer Science & Engineering.

In order to model this, we experiment with a "prescribed" latent structure, which is based on the assumption that courses belonging to the same department provide the same set of knowledge components and that courses belonging to different departments do not share any of their provided knowledge components with each other. In this case, we allocate a distinct set of l latent dimensions for each department. For example, if $l = 5$, and we are working with 10 departments, then the number of dimensions for that approach will be $5 \times 10 = 50$ dimensions. We will refer to the CKRM-based method that uses this prescribed latent structure as **CKRMdep**.

Within that prescribed structure, for each provided knowledge component vector ($\mathbf{p}_c$) we need to estimate only l values, whereas for each required knowledge component vector ($\mathbf{r}_c$), we can potentially be estimating all dimensions.

Textual-based Knowledge Component Space

A source that offers information about the knowledge components provided by courses is their textual descriptions in the University course catalog. These are usually short descriptions of what different knowledge components are provided by the courses in a form of free-text sentences and/or keywords. We hypothesize that it may be possible to derive a knowledge component space using these descriptions.

In order to test this hypothesis, we use the set of 2-ngrams that appear in the textual descriptions of the courses as the knowledge component space and represent each course as a bag-of-ngrams vector. With this representation, we can use the vectors in the knowledge component space as indicator vectors and just estimate the required knowledge component space, or we can estimate the non-zero entries of the provided knowledge component space along with estimating the required knowledge component space. In the latter case, the weights on the provided knowledge component vectors can be viewed as indicating some type of relative importance of the different dimensions

(i.e., ngrams) in that course. We will refer to the CKRM-based method that uses the textual descriptions of courses as **CKRMtext**.

5.3.2 Parameter Estimation

The parameters of the CKRM-based methods are the required knowledge component vectors associated with each course, i.e., the various $\mathbf{r}_c$ vectors, and the provided knowledge component vectors of each course, i.e., the $\mathbf{p}_c$ vectors (the latter vectors are estimated for all the approaches except when using them as indicator vectors in CKRMtext).

We use the squared error loss function to estimate these parameters. For the approaches that estimate the provided knowledge component vectors, the optimization problem is

$$\begin{array}{ll} \underset{\mathbf{b},\mathbf{R},\mathbf{P}}{\text{minimize}} & \frac{1}{2}\sum_{s,c\in\mathbf{G}}(g_{s,c}-\hat{g}_{s,c})^2+\frac{\alpha}{2}\Big(\|\mathbf{R}\|_F^2+\|\mathbf{P}\|_F^2+\|\mathbf{b}\|_2^2\Big) \\ \text{subject to} & \mathbf{b}\geq 0,\mathbf{R}\geq 0,\ \mathbf{P}\geq 0, \end{array} \tag{5.4}$$

where $g_{s,c}$ is the actual grade, $\hat{g}_{s,c}$ is the predicted grade (computed as in Eq. 5.3), $\mathbf{b}\in\mathbb{R}^n$ is the vector of course biases, $\mathbf{R}\in\mathbb{R}^{n\times d}$ is the matrix whose rows are the required knowledge component vectors, $\mathbf{P}\in\mathbb{R}^{n\times d}$ is the matrix whose rows are the provided knowledge component vectors, and α is a regularization parameter to avoid overfitting. The non-negativity constraints on $\mathbf{R}$ and $\mathbf{P}$ are enforced since they represent knowledge acquisition, which should be non-negative. Note that for CKRMdep and CKRMtext, $\mathbf{P}$ has a predefined sparse structure, so only the weights of its encoded non-zero entries are estimated. For CKRMtext that uses the provided knowledge component vectors as indicator vectors, the optimization problem is

$$\begin{array}{ll} \underset{\mathbf{b},\mathbf{R}}{\text{minimize}} & \frac{1}{2}\sum_{s,c\in\mathbf{G}}(g_{s,c}-\hat{g}_{s,c})^2+\frac{\alpha}{2}\Big(\|\mathbf{R}\|_F^2+\|\mathbf{b}\|_2^2\Big) \\ \text{subject to} & \mathbf{b}\geq 0,\mathbf{R}\geq 0. \end{array} \tag{5.5}$$

The optimization problems of Eqs. 5.4 and 5.5 are solved using a Stochastic Gradient Descent (SGD) algorithm, which is an iterative algorithm. Algorithm 1 provides the detailed procedure and gradient update rules. Matrices $\mathbf{R}$ and $\mathbf{P}$ are initialized with small random values as the initial estimate (line 6). In each iteration of SGD (lines 7–27), if the course has at least l courses taken prior to it, then its required knowledge

component vector $\mathbf{r}_c$ is updated as well as the preceding j courses' provided knowledge component vectors $\mathbf{p}_{c_i}$. This process is repeated until the RMSE on the validation set does not decrease further or the number of iterations has reached a predefined threshold. Note that, for solving Eq. 5.5, lines 20–23 are ignored and the non-zero entries of $\mathbf{P}$ are just used as indicator vectors.

Algorithm 1 CKRM:Learn

1: **procedure** CKRM_LEARN
2: $l \leftarrow$ minimum # prior courses
3: $\eta \leftarrow$ learning rate
4: $\alpha \leftarrow$ regularization weight
5: $iter \leftarrow 0$
6: Init $\mathbf{b}$ and the non-zero entries of $\mathbf{R}$ and $\mathbf{P}$ with random values in [-0.001, 0.001]
7: **while** $iter < maxIter$ or RMSE on validation set decreases **do**
8: **for all** $g_{s,c} \in \mathbf{G}$ **do**
9: $j \leftarrow$ # courses taken by s prior to c
10: **if** $j \geq l$ **then**
11: $c_j \leftarrow$ last course taken by s prior to c
12: $\mathbf{k}_{s,j} \leftarrow 0$
13: **for all** $c_i \in \mathbf{g}_s$ s.t. c_i was taken by s prior to c **do**
14: $\mathbf{k}_{s,j} \leftarrow \mathbf{k}_{s,j} + \xi(s, c_j, c_i)\ g_{s,c_i} \cdot \mathbf{p}_{c_i}$
15: **end for**
16: $\hat{g}_{s,c} \leftarrow \mathbf{r}_c\ \mathbf{k}_{s,j}^T$
17: $e_{s,c} \leftarrow g_{s,c} - \hat{g}_{s,c}$
18: $b_c \leftarrow b_c + \eta\ (e_{s,c} - \alpha\ b_c)$
19: $\mathbf{r}_c \leftarrow \mathbf{r}_c + \eta \cdot (e_{s,c} \cdot \mathbf{k}_{s,j} - \alpha \cdot \mathbf{r}_c)$
20:
21: **for all** $c_i \in \mathbf{g}_s$ s.t. c_i was taken by s prior to c **do**
22: $\mathbf{p}_{c_i} \leftarrow \mathbf{p}_{c_i} + \eta \cdot (e_{s,c}\ \xi(s, c_j, c_i)\ g_{s,c_i} \cdot \mathbf{r}_c - \alpha \cdot \mathbf{p}_{c_i})$
23: **end for**
24: **end if**
25: **end for**
26: $iter \leftarrow iter + 1$
27: **end while**
28: **return** $\mathbf{b}$, $\mathbf{R}$ and $\mathbf{P}$
29: **end procedure**

5.4 Experimental Evaluation

5.4.1 Dataset Preprocessing

The data used in our experiments was obtained from the College of Science and Engineering at University of Minnesota and includes 12 degree programs. The data that we used span a period of about 14 years (Fall 2002 to Spring 2015). From that dataset, we extracted the students who were registered at the University for at least three terms[1]. For each of these students, we extracted the set of courses that belong to these 12 majors. We removed any courses that were taken as pass/fail. The initial grades were in the A–F scale, which were converted to the 4–0 scale using the standard letter grade to GPA conversion. The statistics of the extracted majors are shown in Table 5.1.

Table 5.1: Information about the Different Majors

Major	**Abbrev.**	**#Students**	**Flexibility**
Mathematics	MATH	1,198	0.704
Statistics	STAT	340	0.698
Physics	PHYS	319	0.664
Chemistry	CHEM	916	0.653
Computer Science	CSE	1,487	0.609
Electrical Engineering	ECE	856	0.589
Materials Science	MATS	288	0.520
Chemical Engineering	CHEN	999	0.512
Mechanical Engineering	ME	1,620	0.490
Biomedical Engineering	BMEN	750	0.485
Aerospace Engineering	AEM	639	0.439
Civil Engineering	CE	737	0.439

The majors are sorted with respect to their flexibility in a decreasing order (see Section Sec. 5.4.1 for the definition of the major's flexibility).

Table 5.1 also shows each major's *flexibility*, which is a measure that we computed in order to differentiate between degree programs that that have a large number of electives and the students' degree programs tend to include different sets of courses (flexible) over those that offer a few electives and the degree programs of all students are quite similar (restricted). As our results will show, the major's flexibility impacts the performance of certain models. We computed the major's flexibility as the average

[1]There are three terms at this University: Fall, Spring and Summer.

course offering flexibility over all course offerings that belong to that major, weighted by the number of pairs of students in that offering. We computed the flexibility of a course offering c as one minus the average Jaccard coefficient of the courses that were taken by the students that took c prior to taking this class. The flexibility will be low if the students that took c have taken very similar courses before c and high otherwise.

Table 5.2: Datasets Statistics

	Train	**Validation**	**Test**
#Students	84,311	26,606	21,954
#Courses	8,355	3,326	1,708
#Grades	1,423,853	77,616	55,866

These statistics are accumulated over the eight datasets created for the eight test terms (see Sec. 5.4.2).

A course that belongs to some department is usually taken by two sets of students: those who major in that department and those who major in another department. These two sets of students have different background since they belong to different majors. We thus created two instances for each course that is taken by these two sets of students, so as to treat each instance as a unique course.

For CKRMtext, we extracted the keywords from each course description after removing the stopwords and extracted the 2-ngrams that exist within a window of size 3. We then created a binary course-by-ngrams matrix, where each course was represented as a vector of its ngrams, that was used as the provided knowledge component matrix $\mathbf{P}$.

5.4.2 Generating Train, Validation, and Test Sets

The entire dataset was used to extract eight different subsets in order to assess the performance of the different methods. Specifically, we selected the eight most recent Fall and Spring terms in our dataset to predict their grades (which we will refer to as the set of test terms $\mathcal{T}$), where for each of these test terms $t \in \mathcal{T}$, only the terms prior to t are used for training and validation. The training, validation and test sets were extracted as follows. For each test term t, the term prior to it that is either a Fall or

a Spring term (not a Summer term) is used for validation and model selection, and all the terms prior to the validation term are used for learning the model. For a student to be considered in the training set, he/she must have taken at least three courses in the training set. This is to ensure that the students have taken a sufficient number of courses so that CKRM can capture knowledge accumulation. Also, we did not consider a course for predicting its grades in the validation or test set if its required knowledge component vector ($\mathbf{r}_c$) was estimated, during learning the model, less than 50 times, as we considered such courses not to have reliable estimated required knowledge component vectors. Therefore, for a course to be considered for prediction during validation or testing, it must have been taken by at least 50 students after at least 3 courses. The statistics about the accumulated training, validation and test sets over the eight subsets of data are shown in Table 5.2.

Following the row-centering technique used by Polyzou *et al.* [4] that was shown to greatly improve the prediction performance of CSR, we centered each student's grade that exists in the training set around his GPA that is computed using his/her grades in that set. This row centering takes a notion of student bias into account. Specifically, for each student, we computed his/her GPA using his/her grades that exist in the training set and then subtracted each of these grades from his/her GPA. Since these row-centered grades are not restrictively non-negative, we removed the constraint of non-negativity on $\mathbf{R}$ while estimating the parameters of the CKRM-based methods.

5.4.3 Baseline/Competing Methods

In our experiments, we compared the performance of the CKRM-based methods against the following competing methods:

1. **CSR:** This is the course specific regression model that was described in Sec. 3.1.

2. **Matrix Factorization (MF):** This approach predicts the grade for student s in a course c following Eq. 3.1.

3. **BiasOnly:** This method is a special case of MF, in which the number of latent dimensions is 0. That is, it predicts the grade for student s in a specific course c using only the bias terms in Eq. 3.1.

The optimization problems for both MF and BiasOnly methods were solved using an SGD algorithm, which is terminated after 1000 iterations or when the RMSE value on the validation set converges.

5.4.4 Evaluation Methodology and Performance Metrics

We evaluated the performance of the different approaches by using them to predict the grades for each of the eight test terms in our dataset using the data from the terms prior to each test term for training and validation (see Table 5.2).

We evaluated the statistical significance of the results obtained by the different methods using a paired-sample one-tailed t-test. Specifically, we used the ticks percentages of the courses belonging to each major in each of the eight datasets as the data points for each method.

5.4.5 Model Selection

We did an extensive search in the parameter space for model selection. We experimented with the regularization parameter α in the range [1e-5, 0.1] and with the learning rate η in the range [5e-5,1]. For CKRMall and CKRMdep, we used the number of dimensions in the range [10, 50] with a step of 10, whereas for MF we used it in the range [10, 60] with a step of 5. For the CKRM-based methods, we experimented with the parameter λ in the range [0, 1] with a step of 0.1.

The training set was used for estimating the models, whereas the validation set was used to select the best performing parameters in terms of the overall RMSE of the validation set.

5.5 Results

For each of the 12 departments, we divided the results into the set of courses that belong to the student's major (*major* courses) and the set of courses that do not belong to his/her major (*non-major* courses), since these two groups of courses represent different populations.

We organized the experimental results into four parts. The first and second show a quantitative comparison of the CKRM-based methods against each other as well as

against the competing methods on major and non-major courses, respectively. The third one discusses the actual versus predicted letter grade distributions. Finally, the third shows a qualitative analysis on CKRMtext.

5.5.1 Quantitative Performance on Major Courses

Table 5.3 shows the performance achieved by the CKRM-based and competing methods on major in terms of 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 three CKRM-based methods, we can see that their performance is quite similar. If we consider the best performing entries across the different departments and error levels we see that one of them outperforms the other two. However, even when a method does better than another one, the differences are fairly small. The close performance of the three methods was also confirmed by the statistical significance tests that we ran, which showed that the performance difference of the three schemes were not statistically significant for most departments.

Comparing the performance achieved by the CKRM-based methods against that achieved by CSR, we see that the former leads to more accurate predictions and its performance advantage is greater for the flexible majors than the restricted ones. This is further illustrated in Table 5.4, which shows the average points improvements of the CKRM-based methods based on the majors' flexibility. The CKRM-based methods achieve an average improvement of 1.07, 1.01, and 0.91 points over CSR in the four most flexible majors, as opposed to 0.17, -0.14, and -0.04 points in the four least flexible ones for the no error, within one tick, and within two ticks errors, respectively. These improvements also indicate that the CKRM-based methods do considerably better than CSR in terms of the no error predictions. These results confirm our hypothesis that CSRM's performance degrades as the major's flexibility increases, since this method depends on the prior set of courses to predict the grades, which can fail in such flexible majors as each student can take a different combination of courses that offer the same knowledge components required for performing well in that course.

Comparing the performance achieved by the CKRM-based methods against that achieved by MF and BiasOnly, we see that they also outperform both MF and BiasOnly in most cases and that their performance is statistically significant over both baselines

Table 5.3: Prediction Performance of the Different Methods on Major Courses

#Ticks	Method	MATH	STAT	PHYS	CHEM	CSE	ECE
Percentage of grades predicted with no error	BiasOnly	16.24	22.66	24.14	20.57	23.08	23.53
	MF	16.55	21.14	24.71	20.34	23.36	22.66
	CSR	20.24	27.81	31.90	21.85	25.51	23.76
	CKRMdep	19.05	28.95	34.19	23.43	25.84	24.77
	CKRMall	18.65	28.19	30.75	22.90	25.70	24.73
	CKRMtext	19.52	28.19	32.18	22.15	25.31	23.90
Percentage of grades predicted with an error of at most one tick	BiasOnly	48.32	55.04	64.08	54.14	59.04	60.29
	MF	48.32	55.24	63.50	52.86	59.44	58.92
	CSR	54.10	56.57	63.22	57.93	61.04	61.54
	CKRMdep	54.29	57.71	65.51	57.75	60.68	62.09
	CKRMall	54.81	57.71	63.22	57.81	61.25	61.95
	CKRMtext	53.66	57.33	63.51	57.58	60.88	61.54
Percentage of grades predicted with an error of at most two ticks	BiasOnly	75.09	79.04	85.63	77.51	81.03	83.32
	MF	74.97	79.80	85.92	77.39	81.80	82.22
	CSR	76.12	75.61	85.63	79.08	81.66	83.96
	CKRMdep	76.11	77.71	85.63	78.43	81.35	84.24
	CKRMall	76.20	76.95	85.92	79.31	81.80	83.46
	CKRMtext	75.49	78.66	85.35	78.56	81.38	84.29
# Predicted Grades		2,525	525	348	1,716	5,120	2,176

#Ticks	Method	MATS	CHEN	ME	BMEN	AEM	CE
Percentage of grades predicted with no error	BiasOnly	20.51	18.18	24.87	28.53	28.12	22.29
	MF	19.33	17.86	23.64	27.98	26.56	21.90
	CSR	21.87	21.54	25.17	30.97	31.87	22.33
	CKRMdep	22.78	22.92	24.96	30.30	32.43	23.07
	CKRMall	21.33	20.84	25.25	30.05	31.19	22.97
	CKRMtext	22.96	22.41	24.80	30.30	32.31	22.92
Percentage of grades predicted with an error of at most one tick	BiasOnly	51.54	52.75	61.95	72.09	68.36	60.02
	MF	52.36	53.52	60.92	72.08	69.73	59.14
	CSR	57.53	54.61	64.44	74.16	67.91	60.31
	CKRMdep	57.53	55.12	63.70	74.40	66.89	60.07
	CKRMall	56.72	54.41	63.51	74.65	67.40	60.31
	CKRMtext	57.62	54.87	63.38	74.16	67.34	60.51
Percentage of grades predicted with an error of at most two ticks	BiasOnly	75.87	79.00	84.82	92.62	89.45	81.28
	MF	74.77	79.51	84.45	92.30	90.04	81.52
	CSR	79.58	77.53	85.99	92.61	86.56	81.08
	CKRMdep	79.04	77.91	85.41	92.42	86.10	81.52
	CKRMall	79.67	76.15	85.65	92.67	85.99	80.74
	CKRMtext	79.31	77.72	85.77	92.48	86.10	81.23
# Predicted Grades		1,102	3,124	5,601	1,637	1,770	2,046

The majors are sorted in descending order with respect to their flexibility (see Table 5.1). See Sec. 5.4.4 for the definition of a tick. Underlined entries denote the best value obtained for each major for each #ticks.

Table 5.4: Effect of Major's Flexibility on the Relative CKRM's Performance on Major Courses

	Average Points Improvement		
#Ticks	**Most Flexible**	**Flexible**	**Least Flexible**
Baseline	CSR		
No error	1.07	0.95	0.17
Within one tick	1.01	0.34	-0.14(N/A)
Within two ticks	0.91	0.23	-0.04(N/A)
Baseline	Best of MF & BiasOnly		
No error	5.40	2.73	1.81
Within one tick	3.52	2.62	0.62
Within two ticks	0.44	0.79	-0.73(N/A)

The 12 majors are divided into three groups of four majors each, according to their flexibility (see Table 5.1). Each of these points is averaged over the included majors' points improvements. N/A denotes no average improvement over the corresponding baseline(s).

in some cases. As shown in Table 5.4, the CKRM-based methods tend to have greater improvement over MF and BiasOnly in the most flexible and flexible major groups than in the least flexible ones.

Comparing the performance of CSR against that of MF and BiasOnly, we can see that CSR does generally better than both of them.

5.5.2 Quantitative Performance on Non-major Courses

Table 5.5 shows the prediction performance achieved by the CKRM-based and competing methods on the set of non-major courses in terms of 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 three CKRM-based methods, we can see that their performance is quite similar, and there was no statistically significant difference in their performance. Comparing the performance of the CKRM-based methods against that of the competing approaches, we can see that the former lead to more accurate predictions, with CSR being comparable to them. Both MF and BiasOnly tend to have similar performance with the other methods in terms of the percentage of grades predicted with an error of at most two ticks.

Table 5.5: Prediction Performance of the Different Methods on Non-major Students

#Ticks	Method	Non-major Courses
Percentage of grades predicted with no error	BiasOnly	20.55
	MF	20.99
	CSR	25.08
	CKRMdep	24.86
	CKRMall	<u>25.39</u>
	CKRMtext	24.85
Percentage of grades predicted with an error of at most one tick	BiasOnly	56.48
	MF	56.27
	CSR	59.96
	CKRMdep	60.16
	CKRMall	<u>60.28</u>
	CKRMtext	60.07
Percentage of grades predicted with an error of at most two ticks	BiasOnly	80.58
	MF	80.05
	CSR	80.58
	CKRMdep	80.85
	CKRMall	<u>81.02</u>
	CKRMtext	80.70
# Predicted Grades		13,423

Underlined entries denote the best value obtained for each #ticks.

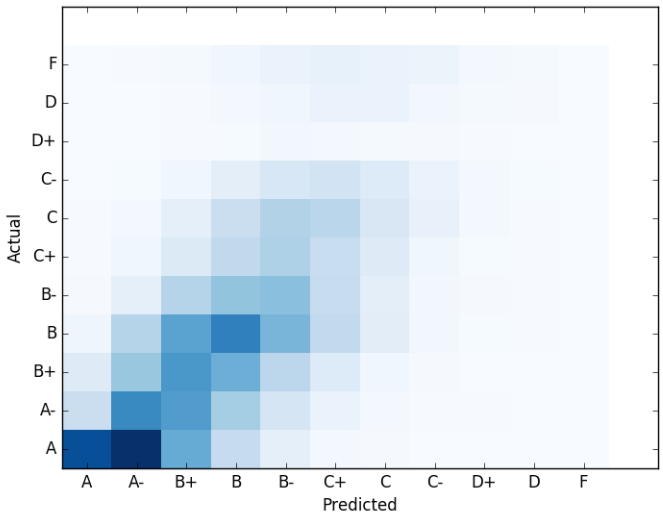


Figure 5.1: Predicted versus Actual Letter Grade Distribution using CKRMdep

5.5.3 Predicted versus Actual Letter Grade Distribution

Fig. 5.1 shows the grade distribution for actual versus predicted grades using CKRMdep. As shown in the figure, the lower grades have higher variation in their predictions than the higher ones. This is further illustrated in Fig. 5.2 that shows the per-actual-letter-grade distribution of the predicted grades. As we go from "A" to "F" grades, the tick errors get larger, with having more over-predictions for the lowest grades. This may be due to having students getting higher grades in courses without acquiring all of their provided knowledge components.

5.5.4 Qualitative Analysis of CKRMtext's Models

The fact that the performance of CKRMtext's models are comparable to that of the other two latent space based variants of CKRM (as discussed in Sec. 5.5.1) is important, because the models estimated by CKRMtext are easier to interpret (since their dimensions correspond to keywords extracted from the course descriptions). As a result, they can be analyzed in order to learn, from students' historical data, the importance of each of the knowledge components for each course.

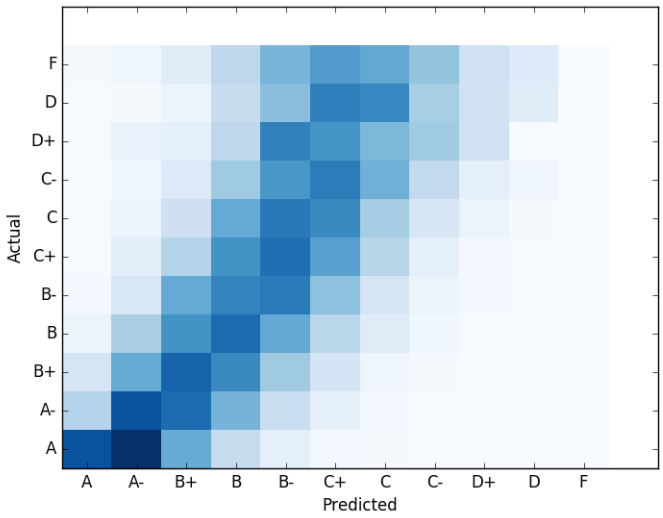


Figure 5.2: Predicted Letter Grade Distribution Per Actual Letter Grade using CKR-Mdep

For this reason, we analyzed the results of CKRMtext's models, as follows. For each course, we extracted, from the students who took that course, the top 2-ngrams that have the highest weights in their knowledge states prior to taking that course (see Eq. 5.1) and computed the percentage of its extracted top ngrams matching the descriptions of the course's pre-requisites[2]. We found that most courses have their top ngrams matching only 0–39% of their pre-requisite descriptions. This suggests that there are other knowledge components not listed in the course's pre-requisite descriptions that also affect the student's performance in that course.

In order to better understand the type of information that these other knowledge components capture, we manually analyzed the top-20 ngrams for the CSE courses. Table 5.6 shows a sample of four of these courses along with their top ngrams. We can see that the ngrams (shown in black) that are not included in the text description of the pre-requisites are also relevant for the requirements of these courses. For

[2]These results were obtained by learning models to estimate the actual grades and not the row-centered grades. This allowed us to have both $\mathbf{R}$ and $\mathbf{P}$ to be non-negative and as such made the results more interpretable.

Table 5.6: Top-20 Keywords for a Sample of Four CSE Courses

CSCI 2011 – Discrete Structures of Computer Science
Top Features: calculus space:15.97, functions polynomials:12.72, quantitative systems:9.76, integration involving:9.48, principles systems:9.21, introduction programming:8.63, language languages:8.26, curves space:8.23, language structures:8.15, data languages:7.8, functions taylor:7.79, calculus integration:7.62, language programming:7.5, data programming:6.44, involving taylor:6.25, forces mechanical:6.24, modularity programming:6.2, languages programming:6.03, development program:5.58, motion systems:5.53
CSCI 4203 – Computer Architecture
Top Features: logical models:6.38, analysis models:4.91, computer machine:4.35, languages models:4.24, mathematical models:2.48, data languages:2.25, computer mathematical:2.17, computer programming:1.98, introduction programming:1.76, probability sampling:1.69, analysis data:1.67, formal models:1.63, computer models:1.61, distributions sampling:1.38, functions methods:1.27, networks programming:1.16, programming projects:1.13, algebra boolean:1.11, communication projects:1.1, development program:1.06.
CSCI 5221 – Foundations of Advanced Networking
Top Features: data programming:3.12, data network:2.94, computer programming:2.21, language structures:1.69, networks programming:1.61, language programming:1.21, architectures routing:1.1, architectures examples:1.06, development program:1.05, computer science:0.95, java object:0.94, network programming:0.92, architectures protocols:0.81, java programming:0.72, communication programming:0.68, architectures network:0.68, computer data:0.64, data networks:0.62, concepts programming:0.6, java oriented:0.54.
CSCI 5512 – Artificial Intelligence II
Top Features: language structures:1.44, computer programming:1.37, data programming:1.24, introduction programming:1.16, control programming:1.16, computer machine:1.13, language programming:1.06, applications sensing:1.04, analysis data:1.01, dynamics kinematics:0.98, java object:0.95, introduction theorem:0.92, applications programming:0.89, based programming:0.87, differential equations:0.85, applications based:0.82, analysis design:0.79, inverse kinematics:0.73, development program:0.72, applications robotics:0.67.

The ngrams colored in red denote those that exist in the course's pre-requisite descriptions. The weight of each ngram is shown next to it, which is computed as explained in Section Sec. 5.5.4.

instance, for the network course (CSCI 5221), there are three ngrams that contain the word "java" ("java object", "java programming" and "java oriented"), along with other ngrams about programming languages in general. This suggests that the students' performance in the programming courses, especially those that taught the Java language had significant impact on their performance in that course. Another example is the Artificial Intelligence course (CSCI 5512), which has eight of its top 20 ngrams, namely "control programming", "applications sensing", "dynamics kinematics", "applications programming", "based programming", "applications based", "inverse kinematics", and "applications robotics", not appearing in the pre-requisites. However, after some further analysis, we determined that these ngrams appear in the description of the CSE course entitled "CSCI 5551, Introduction to Intelligent Robotic Systems", which is not listed as a pre-requisite for that course. This also suggests that students' performance in CSCI 5551 along with the other introductory CSE courses that contain the remaining top ngrams highly affect their performance in CSCI 5512. Similar insights can be gained from the other courses.

This analysis can provide information about the hidden or informal knowledge components whose acquisition by previous students have greatly affected their performance in the target courses. Moreover, these knowledge components can be mapped back to their corresponding courses, which would tell us about the specific courses that have more impact on the performance of students in these courses. This can help in improving the pre-requisite structure and/or the suggested degree plans of the various degree programs in order to take the actual learned structure into account. It can also help in providing future students with the knowledge components (or courses) that have had more impact on the previous students' performance in the different courses, other than the ones listed in the course's pre-requisites.

5.6 Summary

In this chapter, we modeled the next-term grade prediction problem in a traditional University setting as Cumulative Knowledge-based Regression Models (CKRM) that accumulate the performance of a student in all the courses that he/she has previously taken in order to predict his/her future grades. We conducted an extensive experimental

evaluation on a large dataset that includes 12 degree programs of the College of Science & Engineering at University of Minnesota. The results showed that the CKRM-based methods are able to estimate more accurate predictions than the competing methods and some of these improvements are statistically significant. Moreover, the qualitative analysis performed on the CKRM-based methods that use the textual course descriptions showed that they can be used to identify the knowledge required for students to perform well in courses.

Chapter 6

Context-aware Non-linear and Neural Attentive Knowledge-based Models for Grade Prediction

In Chapter 5, we developed Cumulative Knowledge-based Regression Models (CKRM) that build on the idea of accumulating knowledge over time. CKRM predicts the student's grades as the similarity between his/her knowledge state and the target course. Both a student's knowledge state and a target course are represented as low-dimensional embedding vectors and the similarity between them is modeled by their inner product. A student's knowledge state is implicitly computed as a linear combination of the so-called "provided" knowledge component vectors of the previously-taken courses, weighted by his/her grades in them.

In this chapter, we develop context-aware non-linear and neural attentive models that improve upon CKRM from two perspectives. First, they model the complex interactions among prior courses to better estimate a student's knowledge state, by using two different approaches. In the first approach, we hypothesize that each course provides a set of knowledge components at a specific knowledge level. It uses a non-linear model that aggregates the weighted prior course embeddings by employing a maximum-based

pooling layer along each component of the prior courses' embeddings. In the second approach, we hypothesize that prior courses contribute differently towards a target course, and that some of them may not be relevant to it. Motivated by the success of neural attentive networks in different fields [41–46], we learn attention weights for the prior courses that denote their importance to a target course using two different activation functions. The first, called the softmax activation function, is the most commonly-used function, which converts a given input vector of real weights to a probability distribution. The second, called the sparsemax activation function, was recently proposed to truncate the smaller weighted values to zero, hence producing sparse attention weights. This is useful when the input contains some relevant and other irrelevant objects to the object of interest. Second, the proposed models consider the effect of the concurrently-taken courses while predicting a student's grade in a target course. We hypothesize that the knowledge provided by concurrent courses affect the knowledge required by a target course. We model the interaction between the concurrent and target course using non-linear and neural attentive models, as well.

In the following sections, we present the summary of our results in Sec. 6.1. Then, we explain our methods in Sec. 6.2, Sec. 6.3 and Sec. 6.4, describe the experimental setup and evaluation methodology in Sec. 6.5, discuss the results in Sec. 6.6, and summarize the Chapter in Sec. 6.7.

6.1 Main Contributions

The main contributions in this chapter are as follows:

1. We propose context-aware non-linear and neural attentive knowledge-based models for grade prediction that improve upon the linear CKRM model by: (i) using non-linear as well as neural attentive models to capture the complex interactions among prior courses while aggregating their embeddings to compute a student's knowledge state; and (ii) modeling the effect of the concurrently-taken courses using non-linear and neural attentive models. To our knowledge, this is the first work to model the effect of the concurrently-taken courses in grade prediction.

2. We performed an extensive experimental evaluation on a real world dataset obtained from the University of Minnesota that spans a period of 16 years and consists of ∼1.5 grades. The results show that the proposed context-aware non-linear and neural attentive models outperform other competing methods significantly in the prediction accuracy. In addition, they show the effectiveness of estimating a student's knowledge state via a non-linear or neural attentive model over estimating it via a linear model. The results also show that modeling the interaction between the target and concurrent courses help improve the prediction results. A qualitative analysis on the neural attentive models show that the attention weights learned by these models are useful for uncovering the hidden pre-requisites for courses, which can be useful in degree planning and course sequencing.

6.2 Non-linear and Neural Attentive Knowledge-based Models

In Chapter 5, we developed the CKRM method that uses shallow linear models to aggregate the prior courses' embeddings taken by a student in order to estimate his/her knowledge state. We hypothesize that there are more complex interactions among prior courses that cannot be modeled via a linear model. We develop two different approaches to learn these interactions: a non-linear maximum knowledge-based model (Sec. 6.2.1), and a neural attentive knowledge-based model (Sec. 6.2.2).

6.2.1 Maximum Knowledge-based Models

In this section, we develop a **MA**ximum **K**nowledge-based model (MAK), which estimates a student's knowledge state by applying a maximum-based pooling layer on the prior courses. We use CKRM as the underlying model (see Chapter 5).

Motivation

Undergraduate degree programs are structured in a way such that earlier courses provide basic knowledge that is built upon in the later courses that provide more advanced knowledge. Consider the toy example shown in Figure 6.1, which shows the provided

knowledge component vectors for two courses: Introduction to Programming in C/C++ and Advanced Programming Principles. We expect that the introduction to programming course provides basic knowledge to programming to freshman students who may be exposed to programming for the first time. The advanced programming course builds on the knowledge acquired by the introductory course, and provides more advanced knowledge components related to programming principles and programming languages. When a student takes the introductory then the advanced course, he/she can only acquire the maximum knowledge provided by both of them, since each course provides very similar knowledge components, but at a different knowledge level.

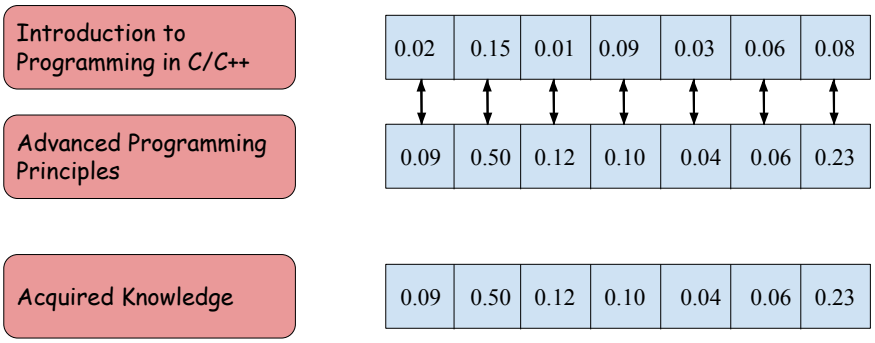


Figure 6.1: Toy example showing the provided knowledge component vectors for two Computer Science courses.

Maximum-based Pooling Layer for Prior Courses

Based on our hypothesis explained above, we can estimate a student s's knowledge state at the beginning of term t as follows:

$$\mathbf{k}_{s,t} = \begin{bmatrix} \max_i \Big(\xi(s, w_{s,i}, t) \ g_{s,i} \ p_{i,1} \Big) \\ . \\ . \\ . \\ \max_i \Big(\xi(s, w_{s,i}, t) \ g_{s,i} \ p_{i,d} \Big) \end{bmatrix}, \forall i \in \mathcal{T}_{s,y} \text{ for } y = 1, \ldots, t-1, \tag{6.1}$$

where $w_{s,i}$ is the relative term number when s took course i, $\xi(s, w_{s,i}, t)$ is a time-based exponential decaying function, $p_{i,z}$ is the zth entry in $\mathbf{p}_i$, $\mathcal{T}_{s,y}$ is the set of courses taken by s in term y, and d is the embedding size of the vector $\mathbf{p}$.

Grade Prediction

Given a student's knowledge state vector, $\mathbf{k}_{s,t}$ (Eq. 6.1) and the required knowledge component vector for a target course j, $\mathbf{r}_j$, we can estimate s's grade in j similar to CKRM as follows:

$$\hat{g}_{s,j} = b_j + \mathbf{k}_{s,t}^T \, \mathbf{r}_j, \tag{6.2}$$

where b_j is a bias term for course j.

6.2.2 Neural Attentive Knowledge-based Models

In this section, we develop a **N**eural **A**ttentive **K**nowledge-based model (NAK), which applies an attention mechanism on prior courses to learn individual weights for them that represent their importance to a target course before aggregating them to estimate a student's knowledge state. We also use CKRM as the underlying model (see Chapter 5).

Motivation

Consider a sample student who is declared in a Computer Science major and is in his/her second or third year in college. Table 6.1 shows the set of prior courses that this student has already take and the set of courses that this student is planning on taking the next term. With CKRM (Chapter 5), all these prior courses would contribute equally to predicting the grade of each target course. However, we can see that, intuitively, from the courses' names, there are courses that are strongly related to each target course and other courses that are irrelevant to it. For instance, it is reasonable to expect that the Intermediate German II course is more related to the Intermediate German I course than any of the other courses that the student has already taken. Along the same lines, we expect that the Algorithms and Data Structures course is more related to other Computer Science courses, such as the Advanced Programming Principles and the Program Design and Development courses. Assuming equal contribution among

these prior courses can hinder the grade prediction model from accurately learning the course representations, and hence lead to poor predictions.

Table 6.1: Sample of prior and target courses for a Computer Science student at the University of Minnesota.

Prior Courses	Target Course
Calculus I, Beginning German, Operating Systems, Intermediate German I, University Writing, Introductory Physics, Peotics in Film, Program Design & Development, Philosophy, Linear Algebra, Internet Programming, Stone Tools to Steam Engines, Advanced Programming Principles, Computer Networks	Intermediate German II
	Probability & Statistics
	Algorithms & Data Structures

Attention-based Pooling Layer for Prior Courses

In order to learn the different contributions of prior courses in estimating a student's grade in a future course, we can employ the CSR technique (see Sec. 3.1) that learns the importance of each prior course in estimating the grade of each future course. Thus, we would estimate a knowledge state vector for each target course j, using the following equation:

$$\mathbf{k}_{s,t,j} = \sum_{w=1}^{t-1} \sum_{i \in \mathcal{T}_{s,w}} \Big(a^p_{i,j} \; g_{s,i} \; \mathbf{p}_i \Big), \tag{6.3}$$

where $a^p_{i,j}$ is a learnable parameter that denotes the attention weight of course i in contributing to student s's knowledge state when predicting his/her grade in course j. However, this solution requires sufficient training data for each (i, j) pair in order to be considered an accurate estimation.

In order to be able to have accurate attention weights between all pairs of prior and target courses, even the ones that do not appear together in the training data, we propose to use the attention mechanism that was recently used in neural networks [42, 50]. The main idea is to estimate the attention weight $a^p_{i,j}$ from the embedding vectors for courses i and j.

In order to compute the similarity between the embeddings of prior course i and

target course j, we use a single-layer perceptron as follows:

$$z_{i,j}^{p} = \mathbf{h}^{pT}\text{RELU}(\mathbf{W}^{p}(\mathbf{q}_i \odot \mathbf{r}_j) + \mathbf{b}^{p}), \tag{6.4}$$

where $\mathbf{q}_i = g_{s,i}\mathbf{p}_i$ denotes the embedding of the prior course i, weighted by s's grade in it, and $\mathbf{W} \in \mathcal{R}^{l\times d}$ and $\mathbf{b} \in \mathcal{R}^{l}$ denote the weight matrix and bias vector that project the input into a hidden layer, respectively, and $\mathbf{h}^{p} \in \mathcal{R}^{l}$ is a vector that projects the hidden layer into an output attention weight, where d and l denote the number of dimensions of the embedding vectors and attention network, respectively. RELU denotes the Rectified Linear Unit activation function that is usually used in neural attentive networks.

Softmax Activation Function The most common activation function used for computing these attention weights is the softmax function [50]. Given a vector of real weights $\mathbf{z}$, the softmax activation function converts it to a probability distribution, which is computed component-wise as follows:

$$\text{softmax}_i(\mathbf{z}) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}. \tag{6.5}$$

Sparsemax Activation Function Although the softmax activation function has been used to design attention mechanisms in many domains [41–46], we believe that using it for grade prediction is not optimal. Since a student enrolls in several courses, and each course requires knowledge from one or a few other courses, we hypothesize that some of the prior courses should have no effect, i.e., zero attention, towards predicting a target course's grade. We thus leverage a recent advance, the sparsemax activation function [48], to learn sparse attention weights. The idea is to define a threshold, below which small probability values are truncated to zero. Let $\triangle^{K-1} := \{\mathbf{x} \in \mathbb{R}^{K} | \mathbf{1}^{T}\mathbf{x} = 1, \mathbf{x} \geq \mathbf{0}\}$ be the $(K-1)$-dimensional simplex. The sparsemax activation function tries to solve the following equation:

$$\text{sparsemax}(\mathbf{z}) = \underset{\mathbf{x}\in K-1}{\text{argmin}} \ \|\mathbf{x} - \mathbf{z}\|^{2}, \tag{6.6}$$

which, in other words, returns the Euclidean projection of the input vector $\mathbf{z}$ onto the probability simplex.

In order to obtain different degrees of sparsity in the attention weights, Laha *et al.* [49] developed a generic probability mapping function for the sparsemax activation function, which they called "sparsegen", and is computed as follows:

$$\text{sparsegen}(\mathbf{z};\gamma) = \text{argmin } \|\mathbf{x}-\mathbf{z}\|^2 - \gamma\|\mathbf{x}\|^2, \tag{6.7}$$

where $\gamma < 1$ controls the L2 regularization strength of $\mathbf{x}$. An equivalent formulation for sparsegen was formed as:

$$\text{sparsegen}(\mathbf{z};\gamma) = \text{sparsemax}\big(\frac{\mathbf{z}}{1-\gamma}\big), \tag{6.8}$$

which, in other words, applies a temperature parameter to the original sparsemax function. Varying this temperature parameter can change the degree of sparsity in the output variables. By setting $\gamma = 0$, sparsegen becomes equivalent to sparsemax.

Grade Prediction

Given a student's knowledge state vector, $\mathbf{k}_{s,t,j}$ (Eq. 6.3) and the required knowledge component vector for a target course j, $\mathbf{r}_j$, we can estimate s's grade in j similar to MAK as follows:

$$\hat{g}_{s,j} = b_j + \mathbf{k}_{s,t,j}^T\ \mathbf{r}_j, \tag{6.9}$$

where b_j is a bias term for course j.

6.3 Context-aware Non-linear and Neural Attentive Models

Another limitation of CKRM and other previous grade prediction methods is that they ignore the effect of concurrently-taken courses into account. We hypothesize that the knowledge provided by concurrent courses can affect the knowledge required by a target course. We estimate a context-aware embedding for a target course that we would like to predict a student's grade in, given the courses taken concurrently with it. We utilize the proposed MAK and NAK models (Sec. 6.2) as our underlying models.

To model the interactions between a target course and other courses taken concurrently with it, we estimate a context-aware embedding for that target course as follows:

$$\mathbf{e}_{j,w} = \mathbf{x}_{j,w} \odot \mathbf{r}_j, \tag{6.10}$$

where $\mathbf{x}_{j,w}$ denotes the aggregated embedding of the courses that are taken concurrently with j in term w, $\odot$ denotes the Hadamard product, and $\mathbf{r}_j$ denotes the required knowledge component vector for target course j. To aggregate the concurrent courses' embeddings, we use non-linear and neural attentive models similar to the ones developed in Sec. 6.2.1 and Sec. 6.2.2, respectively.

6.3.1 Context-aware Maximum Knowledge-based Models

In this section, we develop a **C**ontext-aware **MA**ximum **K**nowledge-based model (CMAK), which models the interactions between a target and concurrent courses using MAK (Sec. 6.2.2) as the underlying model.

The aggregated embedding of the courses that are taken concurrently with j in term w is estimated by applying a maximum-based pooling layer on them, similar to how we aggregated the prior courses' embeddings for the MAK model (Sec. 6.2.1), and is computed as:

$$\mathbf{x}_{j,t} = \begin{bmatrix} \max\limits_i p_{i,1} \\ . \\ . \\ . \\ \max\limits_i p_{i,d} \end{bmatrix}, \forall i \in \mathcal{T}_{\{s,t\}\setminus\{j\}} \tag{6.11}$$

Grade Prediction

Given a student's knowledge state vector, $\mathbf{k}_{s,t}$ (Eq. 6.3) and the context-aware embedding for a target course j, $\mathbf{e}_{j,t}$ (computed using Eq. 6.11), we can estimate s's grade in j as follows:

$$\hat{g}_{s,j} = b_c + \mathbf{k}_{s,t}^T \, \mathbf{e}_{j,t}. \tag{6.12}$$

6.3.2 Context-aware Neural Attentive Knowledge-based Models

In this section, we develop a **C**ontext-aware **N**eural **A**ttentive **K**nowledge-based model (CNAK), which models the interactions between a target and concurrent courses using NAK (Sec. 6.2.2) as the underlying model.

To aggregate the concurrent courses' embeddings, we employ an attention mechanism on them to learn the different contributions of each of them towards the target course, similar to how we aggregated the prior courses' embeddings for the NAK model (Sec. 6.2.2). The aggregated embedding of the courses that are taken concurrently with j in term w is computed as:

$$\mathbf{x}_{j,w} = \sum_{i \in \mathcal{T}_{\{s,w\} \setminus \{j\}}} a^x_{i,j} \mathbf{p}_i, \tag{6.13}$$

where $a^x_{j,t}$ is the attention weight for the concurrent course j, and can be computed using the softmax (Eq. 6.5) or sparsegen (Eq. 6.8) activation function. Note that we use the same embedding vector $\mathbf{p}_i$ for representing both a prior and a concurrent course. The affinity between concurrent course i and target course j is computed in a similar way as in Eq. 6.4, i.e.,

$$z^x_{i,j} = \mathbf{h}^{xT} \text{RELU}(\mathbf{W}^x(\mathbf{p}_i \odot \mathbf{r}_j) + \mathbf{b}^x). \tag{6.14}$$

Grade Prediction

Given a student's knowledge state vector, $\mathbf{k}_{s,t,j}$ (Eq. 6.3) and the context-aware embedding for a target course j, $\mathbf{e}_{j,t}$ (computed using Eq. 6.13), we can estimate s's grade in j as follows:

$$\hat{g}_{s,j} = b_c + \mathbf{k}^T_{s,t,j}\ \mathbf{e}_{j,t}. \tag{6.15}$$

6.4 Model Optimization

We use the mean squared error (MSE) loss function to estimate the parameters of all our proposed models. We minimize the following regularized MSE loss:

$$L = -\frac{1}{2N} \sum_{s,c \in \mathbf{G}} (g_{s,c} - \hat{g}_{s,c})^2 + \lambda ||\Theta||^2, \tag{6.16}$$

where N is the number of grades in $\mathbf{G}$. The hyper-parameter λ controls the strength of L2 regularization to prevent overfitting, and $\Theta = \{\{\mathbf{b}\}, \{\mathbf{p}_i\}, \{\mathbf{r}_i\}\}$ denotes the learnable parameters for the MAK and CMAK models, $\Theta = \{\{\mathbf{b}\}, \{\mathbf{p}_i\}, \{\mathbf{r}_i\}, \mathbf{W}^p, \mathbf{b}^p, \mathbf{h}^p\}$, denotes the learnable parameters for the NAK model, and $\Theta = \{\{\mathbf{b}\}, \{\mathbf{p}_i\}, \{\mathbf{r}_i\}, \mathbf{W}^p, \mathbf{b}^p, \mathbf{h}^p, \mathbf{W}^x, \mathbf{b}^x, \mathbf{h}^x\}$ denotes the learnable parameters for the CNAK model, where $\mathbf{W}^p$, $\mathbf{b}^p$, and $\mathbf{h}^p$ denote the attention mechanism parameters for the prior courses, and $\mathbf{W}^x$, $\mathbf{b}^x$, and $\mathbf{h}^x$ denote the attention mechanism parameters for the concurrent courses.

The optimization problem is solved using AdaGrad algorithm [51], which applies an adaptive learning rate for each parameter. It randomly draws mini-batches of a given size from the training data and updates the related model parameters.

6.5 Evaluation Methodology

6.5.1 Dataset

The data used in our experiments was obtained from a large public university, the University of Minnesota, that includes 96 majors from 10 different colleges, and spans the years 2002 to 2017. At the University of Minnesota, the letter grading system used is A–F, which is converted to the 4–0 scale using the standard letter grade to GPA conversion. We row-centered the student's grades in each term around his/her GPA achieved in previous terms, which was shown to significantly improve the prediction performance in [4]. We removed any grades that were taken as pass/fail. The final dataset includes $54,269$ students, $5,824$ courses, and $1,561,145$ grades in total.

6.5.2 Generating Training, Validation and Test Sets

At the University of Minnesota, there are three terms, Fall, Summer and Spring. We used the data from 2002 to Spring 2015 (inclusive) as the training set, the data from Spring 2016 to Fall 2016 (inclusive) as the validation set, and the data from Summer 2016 to Summer 2017 (inclusive) as the test set. For a target course taken by a student to be predicted, that student must have taken at least four courses prior to the target course, in order to have sufficient data to compute a student's knowledge state vector. We excluded any courses that do not appear in the training set from the validation and test sets.

6.5.3 Baseline Methods

We compared the performance of the proposed models against the following grade prediction methods:

1. **Matrix Factorization (MF):** This method predicts the grade for student s in course i as:

$$\hat{g}_{s,i} = \mu + sb_s + b_i + \mathbf{u}_s^T \mathbf{v}_i, \tag{6.17}$$

   where μ, sb_s and b_i are the global, student and course bias terms, respectively, and $\mathbf{u}_s$ and $\mathbf{v}_i$ are the student and course latent vectors, respectively. We used the squared loss function with L2 regularization to estimate this model.

2. **KRM(sum):** This is the CKRM method proposed in Chapter 5.

3. **KRM(avg):** This is similar to the KRM(sum) method, except that the prior courses' embeddings are aggregated with mean pooling instead of summation. It was shown in later studies, e.g. [52], that it performs better than KRM(sum).

We implemented KRM(sum) and KRM(avg) with a neural network architecture and optimization similar to that of our proposed models.

6.5.4 Model Selection

We performed an extensive search on the parameters of the proposed and baseline models to find the set of parameters that gives us the best performance for each model.

For all proposed and competing models, the following parameters were used. The number of latent dimensions for course embeddings was chosen from the set of values: {8, 16, 32}. The L2 regularization parameter was chosen from the values: {1e-5, 1e-7, 1e-3}. Finally, the learning rate was chosen from the values: {0.0007, 0.001, 0.003, 0.005, 0.007}. For the proposed NAK and CNAK models, the number of latent dimensions for the MLP attention mechanism was selected in the range [1, 4]. For the sparsegen activation function in NAK and CNAK, the L2 regularization parameter γ was chosen from the values: {0.5, 0.9}. For KRM(sum), KRM(avg), MAK and CMAK, the time-decaying parameter λ was chosen from the set of values: {0, 0.3, 0.5, 0.7, 1.0}.

The training set was used for estimating the models, whereas the validation set was used to select the best performing parameters in terms of the overall MSE of the validation set.

6.6 Experimental Results

We present the results of our experiments to answer the following questions:

RQ1. How do the proposed models compare against the competing methods?

RQ2. What is the impact of estimating a student's knowledge state via a non-linear or neural attentive model?

RQ3. What is the impact of modeling the effect of concurrent courses on a student's performance in a target course?

RQ4. Are we able to derive any insights about the importance of different prior courses to target courses from the neural attentive, i.e., NAK, model?

6.6.1 Performance against Competing Methods

Table 6.2 shows the performance of the proposed models against the competing models (**RQ1**). Among the baseline methods, both KRM(sum) and KRM(avg) outperforms MF. KRM(avg) outperforms KRM(sum) in predicting grades within no and one tick errors. Among all competing and proposed methods, the proposed CMAK and CNAK models outperform all baseline methods, with statistically significant improvements in

Table 6.2: Comparison with baseline methods.

Model	Parameters					RMSE (↓)	PTA0 (↑)	PTA1 (↑)	PTA2 (↑)
MF	16	1E-04	1E-02	–	–	0.724	25.7	58.6	79.5
KRM(sum)	32	1E-07	7E-04	0.3	–	0.584	32.6	70.1	87.7
KRM(avg)	32	1E-07	7E-04	0.0	–	0.584	34.9	70.6	87.7
CMAK	32	1E-07	1E-03	0.0	–	0.548† (6.2)	35.1 (0.6)	73.4 (4.0)	89.8 (2.4)
CNAK	32	1E-07	1E-03	1	0.5	0.569† (2.6)	35.5† (1.7)	72.0 (2.0)	88.7 (1.1)

The Parameters columns denote the following model parameters that were selected: for MF, the parameters are: the number of latent dimensions, the L2 regularization parameter, and the learning rate; for KRM(sum), KRM(avg), and CMAK, the parameters are: the embedding size for courses, the L2 regularization parameter, the learning rate, and the time-decaying parameter λ; and for CNAK, the parameters are: the embedding size for courses, the L2 regularization parameter, the learning rate, the number of latent dimensions for the MLP attention mechanism, and the L2 regularization parameter γ for the sparsegen activation function. Underlined entries represent the best performance in each metric. † denotes statistical significance over the best baseline model, using the Student's t-test with a p-level < 0.05. Numbers in parentheses denote the percentage of improvement over the best baseline value in each metric.

Table 6.3: Severe under- and over-predictions by baseline and proposed models.

Model	Severe Under-predictions (↓)	Severe Over-predictions (↓)
KRM(sum)	5.4	6.9
KRM(avg)	5.6	6.7
CMAK	4.9	6.4
CNAK	3.9	6.3

Severe under-predictions denote the percentage of grades that were predicted with three or more ticks lower than the actual grades, while severe over-predictions denote the percentage of grades that were predicted with three or more ticks higher than the actual grades. Underlined entries represent the best performance in each metric.

some metrics, namely the RMSE and PTA0 metrics. CMAK and CNAK achieve 6.2% and 2.6% lower (better) RMSE, and 2.4% and 1.1% more accurate predictions within two tick errors, respectively, than the best performing baseline method. This shows the effectiveness of the proposed context-aware non-linear and neural attentive models in accurately predicting the grades of students in their future courses than all competing methods. Comparing CMAK with CNAK, we see that CMAK outperforms CNAK, achieving 3.7% lower RMSE, and 1.2% more accurate predictions within two tick errors.

Table 6.3 shows the percentage of severe under- and over-predictions that were made by the different baseline and proposed models, denoting the grades that were predicted with three or more tick errors lower and higher than the actual grades, respectively.

Severe under-predictions can result in an opportunity loss for students, urging them not to take these under-predicted courses in fear of lowering their GPAs. Severe over-predictions can result in urging them to take these over-predicted courses that they may not be well-prepared for and may lower their GPAs. For the severe under-predictions, both CMAK and CNAK outperform the KRM variants significantly, achieving 27% and 9% less severe under-predictions. For the severe over-predictions, both CMAK and CNAK also outperform the KRM variants, achieving 5% and 3% less severe over-predictions. Comparing CMAK with CNAK, we see that CMAK outperforms CNAK, achieving 20% less severe under-predictions, and 2% less severe over-predictions. Severe over-predictions usually denote the D and F grades that get high predicted grades. Since the grades in the data are row-centered around the students' average grades and a course bias term is learned for each course, it is hard for all these models to prevent severe over-predictions from occurring.

Table 6.4: Effect of estimating students' knowledge states via non-linear and neural attentive models.

Model	Parameters					RMSE (↓)	PTA0 (↑)	PTA1 (↑)	PTA2 (↑)
KRM(sum)	32	1E-07	7E-04	0.3	–	0.584	32.6	70.1	87.7
KRM(avg)	32	1E-07	7E-04	0.0	–	0.584	34.9	70.6	87.7
MAK	32	1E-07	3E-03	0.0	–	0.571† (2.2)	34.7 (-0.6)	72.1 (2.1)	88.8† (1.3)
NAK(soft)	32	1E-07	7E-04	3	–	0.589 (-0.9)	35.3 (1.1)	71.8 (1.7)	88.0 (0.3)
NAK(sparse)	32	1E-07	7E-04	4	0.5	0.574† (1.7)	35.3† (1.1)	72.1 (2.1)	88.7† (1.1)

The Parameters columns denote the following model parameters that were selected: for KRM(sum), KRM(avg) and MAK, the parameters are: the embedding size for courses, the L2 regularization parameter, the learning rate, and the time-decaying parameter λ; and for the NAK models, the parameters are: the embedding size for courses, the L2 regularization parameter, the learning rate, the number of latent dimensions for the MLP attention mechanism, and the L2 regularization parameter γ for the sparsegen activation function. Underlined entries represent the best performance in each metric. † denotes statistical significance over the best baseline model, using the Student's t-test with a p-level < 0.5. Numbers in parentheses denote the percentage of improvement over the best baseline value in each metric.

6.6.2 Effect of Estimating Student's Knowledge State via Non-linear and Neural Attentive Models

Table 6.4 shows the effect of estimating a student's knowledge state via a non-linear, i.e., MAK, or neural attentive, i.e., NAK, model (**RQ2**). Both the MAK and NAK models outperform the KRM variants, with some statistically significant improvements, showing

the effect of modeling the complex interactions among prior courses when estimating a student' knowledge state. Using a maximum-based pooling layer (MAK) outperforms using an attention-based pooling layer (NAK), implying that the former makes less severe errors in predicting the grades.

Comparing the NAK models with the softmax and sparsemax activation functions, we can see that learning sparse attention weights outperforms learning soft attention weights. This is expected, since not all prior courses are relevant to a target course, as illustrated later in the qualitative analysis in Sec. 6.6.4.

6.6.3 Effect of Modeling Concurrent Courses' Effect

Table 6.5 shows the impact of modeling the concurrent courses' effect on a student's performance in a target course (**RQ3**), using both the MAK and NAK models. CMAK outperforms MAK significantly, achieving 4% lower RMSE, and 1.1% more accurate predictions within two tick errors. On the other hand, CNAK slightly outperforms NAK with 0.9% lower RMSE, and achieves the same percentage of accurate predictions within two tick errors. This shows that modeling the interactions between a target course and concurrent courses helps in improving the prediction accuracy for a student's grade in that target course.

Table 6.5: Effect of modeling concurrent courses on students' performance in target courses.

Model	Parameters					RMSE (↓)	PTA0 (↑)	PTA1 (↑)	PTA2 (↑)
MAK	32	1E-07	3E-03	0.0	–	0.571	34.7	72.1	88.8
CMAK	32	1E-07	1E-03	0.0	–	<u>0.548</u>† (4.0)	35.1† (1.2)	<u>73.4</u>† (1.8)	<u>89.8</u> (1.1)
NAK(sparse)	32	1E-07	7E-04	4	0.5	0.574	35.3	72.1	88.7
CNAK	32	1E-07	1E-03	1	0.5	0.569† (0.9)	<u>35.5</u> (0.6)	72.0 (-0.1)	88.7 (0.0)

The Parameters columns denote the following model parameters that were selected: for MAK and CMAK, the parameters are: the embedding size for courses, the L2 regularization parameter, the learning rate, and the time-decaying parameter λ; and for MAK and CMAK, the parameters are: the embedding size for courses, the L2 regularization parameter, the learning rate, the number of latent dimensions for the MLP attention mechanism, and the L2 regularization parameter γ for the sparsegen activation function. Underlined entries represent the best performance in each metric. † denotes statistical significance over the corresponding non-context-aware model, while using the Student's t-test with a p-level < 0.5.

6.6.4 Qualitative Analysis on the Prior Courses Attention Weights

In this section, we study the behavior of the attention mechanism on prior courses in the NAK model (**RQ4**). Recall the motivational example for the Computer Science student, discussed in Sec. 6.2.2. This student had a set of prior courses and three target courses that we would like to predict his/her grades in (See Table 6.1). Using KRM(sum) or KRM(avg), all the prior courses would contribute equally to the prediction of each target course. Using our proposed NAK(sparse) model, the attention weights for the prior courses with each target course are shown in Table 6.6[1].

We can see that, using the sparsegen activation function, only a few prior courses are selected with non-zero attention weights, which are the most relevant to each target course.

For the Intermediate German II course, we can see that the student's grade in it is most affected by two courses: the Intermediate German I course, and the University Writing course. The Intermediate German I course is listed as a pre-requisite course for the Intermediate German II course. Though the University Writing course is not listed as a pre-requisite course, after further analysis, we found out that the Intermediate German II course requires process-writing essays and are considered part of the grading system. Though the German courses are not part of the student's degree program, and are taken by a small percentage of Computer Science students, our NAK model was able to learn accurate attention weights for them.

The other two target courses, Probability and Statistics, and Algorithms and Data Structures, have totally different prior courses with the largest attention weights, which are more related to them.

These results illustrate that the proposed NAK model was able to uncover the listed as well as the hidden/informal pre-requisite courses without any supervision given to the model.

[1]These results were obtained by learning NAK models to estimate the actual grades and not the row-centered grades. Also, we used $\mathbf{q}_i = \mathbf{p}_i$ in Eq. 6.4. This allowed us to get more interpretable results.

Table 6.6: The attention weights of the prior courses with each target course for the sample student from Table 6.1.

Prior Courses	Target Course
Intermediate German I: 0.6980, University Writing: 0.3020	Intermediate German II
Calculus I: 0.4737, Physics: 0.3794, Program Design & Development: 0.0717, Operating Systems: 0.0497, Computer Networks: 0.0255	Probability & Statistics
Operating Systems: 0.2927, Advanced Programming Principles: 0.2582, Linear Algebra: 0.2313, Physics: 0.2178	Algorithms & Data Structures

Prior courses are sorted in non-increasing order wrt to their attention weights with each target courses for clarity purposes.

6.7 Summary

In this work, we presented context-aware non-linear and neural attentive models that improve upon the previously developed CKRM method (Chapter 5), by: (i) modeling the complex interactions among prior courses for better estimating a student's knowledge state; and (ii) modeling the interactions between a target course and concurrently-taken courses. The experiments showed that the proposed models significantly outperformed all baseline methods. In addition, the proposed neural attentive models are able to capture the listed as well as the hidden pre-requisite courses for the target courses, which can be better used to design better degree plans.

Chapter 7

A Study on Degree Planning and Its Relationship with Graduation GPA and Time To Degree

Several course recommendation methods have been proposed in previous studies. These methods are based on: association rule mining [17], student-based collaborative filtering [16], group popularity ranking [16], content-based recommendation [53], and matrix factorization [16, 54]. Other methods focused on recommending the whole sequence of courses that satisfy the degree requirements [20–22]. These previous methods train their models on all of the past students' registration data, regardless of their graduation GPA and Time To Degree (TTD). A few other studies that developed course recommendation methods have shown through the analysis of their developed methods' results that different GPA-based groups of students tend to follow different sequencing for courses. For instance, Cucuringu *et al.* [55] applied multiple rank aggregation methods, such as PageRank and SVD-Rank, on Math major students at their university to obtain global course sequences that are most consistent with the given data. Their results showed that different GPA-based groups of students tend to follow different course sequencing.

Another line of research focused on developing predictive models for estimating time-to-degree and causal models for understanding the effects of different features on time-to-degree [56–60]. Features like: family background, demographic data, financial aid,

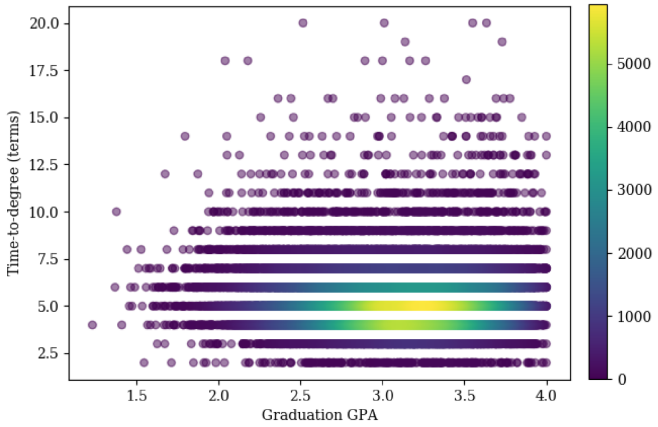


Figure 7.1: Distribution of graduation GPA vs time to degree across 25 different majors (see Section 7.2.1).

on- and off-campus work, course grades, pre-collegiate experience, on- and off-campus course experiences, credit hours, class attendance, and faculty teaching courses taken have all been used in these previous studies to predict or study their effect on whether the student will graduate on-time or over-time. None of these studies have explored the effect of the degree plan itself pursued by students and whether the timing of taking courses and their ordering correlate to time to degree.

To illustrate the differences in the students' academic outcomes, we plotted the distribution of the students' graduation GPA and TTD at the University of Minnesota, which is shown in Figure 7.1. As shown in the figure, there is a large variability in the graduation GPA and TTD of students, with graduation GPAs in the range [1.8, 4.0], and TTD in the range and [3, 17] terms. This suggests that not all previous degree plans should be treated equally for learning good registration patterns. Despite this large variability in the students' graduation GPA and TTD, analyzing the actual students' degree plans and how/if they relate to their academic performance and TTD has received limited attention. We believe that this analysis should provide good insights about how to best utilize the past degree plans to help promote academic success for current and future students.

In this chapter, we provide an analysis of the degree plans taken by past students and their relationship with their academic performance in terms of graduation GPA and

TTD. We try to answer the two following research questions:

RQ1. How does the timing of taking courses with respect to the student's academic level relate to their GPA and TTD?

RQ2. How does the pairwise degree similarity between pairs of students relate to the similarity in their GPA and TTD?

We use a large-scale dataset that consists of 25 majors that have the highest population of degrees granted from different colleges at the University of Minnesota that spans 16 years.

Based on the results of our analysis, it is important for data-driven approaches that utilize student's degree plans, such as course recommendation, course sequence recommendation, and curriculum designing, to: (i) take the graduation GPA and TTD into account when training their models on students' degree plans; (ii) consider the student's academic level when recommending to them a set of courses, and making sure the courses are well-aligned with their academic level; and (iii) account for the student's expected grades and TTD in each course that they recommend. We believe that this can further improve the performance of these methods, especially for marginalized students who struggle with course selection and curriculum planning, and help them towards better academic performance and successful graduation.

In the following sections, we present the summary of our results in Sec. 7.1. Then, we explain our data and metrics used for the analysis in Sec. 7.2, discuss the results of our analysis in Sec. 7.3, present a case study on time to degree prediction using course timing and ordering features in Sec. 7.4, and summarize the Chapter in Sec. 7.5.

7.1 Main Contributions

The main contributions in this chapter are as follows:

1. We perform a large-scale analysis on the degree plans that belong to 25 majors from different colleges on how course timing, i.e., when students take courses with respect to their academic level, and course sequencing relate to the students' GPAs and time to degree.

2. We define several metrics for measuring course timing and similarity of course sequencing between pairs of degree plans. These metrics are then used to derive insights about how degree planning is related to students' GPAs and time to degree.

3. Our analysis shows that: (i) low TTD students tend to take more courses ahead of time, and follow more similar sequencing for the common courses (especially in their later years), than high TTD students; and (ii) low GPA students tend to take more courses ahead of time, and follow more diverse sequencing for the common courses, than high GPA students.

4. We perform a case study that tries to predict whether the student at each semester will graduate on-time or over-time, by using features related to course timing and ordering as pursued by that student. TTD prediction has been explored in several previous studies [56–60], where they used features about student's demographic information, family background, financial aid, on- and off-campus work and experiences, as well as course grades and credit hours. We train several binary classification models using the proposed course timing and ordering features and show that curriculum planning is also a good indicator for TTD prediction.

7.2 Analysis of Degree Planning

7.2.1 Data Extraction and Pre-processing

The data used in our study was obtained from the University of Minnesota, where it spans a period of 16 years (Fall 2002 to Spring 2017). We extracted the set of students who have already completed their degrees on or before Spring 2017. We selected the degree programs that have at least 1,000 graduated students, which accounted for 25 majors from different colleges. Since our study focuses on the timing of courses and their ordering, we focused our study on full-time students and filtered out students who have been enrolled on a part-time basis for more than two terms. In addition, we removed rare courses that were taken by less than 20 students. The statistics for the final dataset used in our analysis is shown in Table 7.1.

We define time-to-degree (TTD) as the actual number of Fall and Spring terms taken

Table 7.1: Dataset statistics.

Major	Students	Courses	Grades
Accounting	848	882	39,996
Art	740	1,461	27,132
Biology	1,311	1,399	53,885
Business & Marketing	738	1,061	33,259
Chemical Engineering	753	742	36,004
Civil Engineering	785	727	33,186
Communication Studies	1,919	2,041	76,504
Computer Science	993	1,011	43,593
Economics	914	1,248	39,059
Electrical Engineering	884	697	37,370
Elementary Education	932	903	37,783
English	1,564	2,144	59,462
Family Social Science	841	976	30,788
Finance	1,194	1,104	56,547
Global Studies	966	1,844	35,942
History	1,055	1,867	40,508
Journalism	2,467	2,256	104,757
Kinesiology	1,117	1,100	57,086
Marketing	1,179	1,157	52,365
Mechanical Engr.	1,266	820	52,786
Nursing	785	794	39,875
Political Science	2,046	2,400	76,296
Psychology	2,688	2,578	104,206
Soc of Law Criminol Devianc	727	1,266	28,253
Spanish Studies	789	1,710	34,365
Total	29,501	34,188	1,231,007

by the student, divided by two. Since the number of students who transferred credits from other institutions or transferred credits from high school constitutes about two thirds of all students on average over all majors, we included them in our analysis by computing their TTD as the sum of their TTD at the University of Minnesota and the estimated number of terms for taking the transfer credits, which we refer to as transfer terms. We estimated the number of transfer terms as follows. For each student that have transferred credits from another college or from high school, let c and x be the number of transfer credits and the maximum number of credits taken by that student in the Fall or Spring terms, respectively. The number of transfer terms is then estimated by dividing c by x.

7.2.2 Data Analysis

Our two research questions focus on studying how the student's academic level when they take their courses as well as the pairwise degree similarity between pairs of students

relate to their graduation GPA and TTD. To address these two questions, we define two sets of metrics: course timing and degree similarity metrics.

Course Timing Metrics

In each department, courses can be taken by students of different academic levels, e.g., freshman or sophomore. Previous studies, such as [52], showed that the student academic level plays an important role in accurately predicting their grades in a future course, since students of the same academic level tend to have similar academic maturity, experience, and knowledge. Based on that, we assume that each course needs to be taken in its corresponding level, which is based on the majority population of students of the same major who previously took this course. To address our first research question, which focuses on the timing of courses and how it relates to the student's academic performance, we measure the difference between the student's academic level when they took a course and the course's derived academic level. We also measure the difference between the academic level of pairs of students who took the same course. Let slevel(s_i, x) be a function that returns the classification code for student s_i (1 for freshman, 2 for sophomore, 3 for juniors and 4 for seniors), when they took course x. And let clevel(x) be a function that returns the derived level for course x that belongs to a specific major, which we compute as the majority student population's level that belong to that major when they took course x. For instance, for a course CSCI 541, if the overall distribution of the students' academic levels when they took it is: 60% seniors, 30% juniors, and 10% sophomores, then clevel(CSCI541) will return the classification code for seniors, which is 4[1]. Note that we only considered courses whose majority population is at least 60% of their whole population. We define two different metrics for computing course timing as taken by students, as follows:

1. **Student-to-course Absolute Level Difference:** Given a course x taken by student s_i, we compute the absolute deviation of s's academic level when they took x from x's academic level as:

$$\text{diff}(s_i, x) = |\text{clevel}(x) - \text{slevel}(s_i, x)|, \tag{7.1}$$

[1]Though this is a simple way to define the course's academic level, it serves as a good starting metric. We plan to investigate other ways of deriving the course's academic level more efficiently in the future.

We will refer to this metric as *Student-to-course Absolute Level Difference*. This metric gives a value in the range [0, 3]. Computing the average of dist(s_i, x) over different student groups tells us how often students in each group take courses at their right academic level, where the smaller the average value is, the more courses that students take at a closer course academic level to their academic level.

2. **Student-to-course Signed Level Difference:** Eq. 7.1 measures the absolute deviation of the student's academic level to the course's derived level, but it does not take into consideration the sign of that deviation. Since a student can take a course ahead or behind its derived level, we need another metric that considers this difference. This will show when different students tend to take their courses. We thus define our next metric, which we will refer to as *Student-to-course Signed Level Difference*, and is computed as:

$$\text{diff}(s_i, x) = \text{clevel}(x) - \text{slevel}(s_i, x), \tag{7.2}$$

This metric gives a value in the range [-3, 3]. Computing the average of this metric over all the courses taken by a student can tell us how often that student tends to take courses at different derived course level from their academic level when taking these courses. The higher the negative direction of this average value, the more lower-level courses the student took , while the higher the positive direction of this average value, the more higher-level courses the student took.

Degree Similarity Metrics

Our second research question focuses on the pairwise degree similarity between pairs of students and it relates to the similarity in their graduation GPA and TTD. To address this research question, we define three different metrics that compute the similarity between a pair of degree plans, as follows.

1. **Student-to-student Course Time Difference:** For each pair of students, we compute the academic level difference when they took the common courses. We will refer to this metric as the *Student-to-student Course Time Difference*, and we

compute it as:

$$\text{diff}(s_1, s_2, x) = |\text{slevel}(s_1, x) - \text{slevel}(s_2, x)|, \tag{7.3}$$

The average of Student-to-student Course Time Difference over all the common courses taken by a pair of students will be low for pairs of students who take the common courses at similar academic levels, and will be high otherwise.

2. **Bag Similarity:** The similarity between two degree plans with respect to the set of courses taken in both of them can be measured by using the Jaccard similarity coefficient between them, which we will refer to as the bag similarity, and is computed as:

$$\text{sim}(d_1, d_2) = \frac{|C_1 \cap C_2|}{|C_1 \cup C_2|}, \tag{7.4}$$

where: C_i is the set of courses taken in degree i. This metric gives us an overall idea about the percentage of courses that are taken in common in the two degree plans.

3. **Sequence Similarity:** The bag similarity metric defined above cannot tell us any information about the ordering of common courses in a pair of degree plans, which can be an important factor for academic performance. Since each course provides specific knowledge that can be useful for performing well in another course, the ordering of courses can affect the student's grades as well as their TTD. Therefore, we define another metric that can tell us how the course sequencing in the two plans aligns with each other. We will refer to this metric as sequence similarity, which we compute as:

$$\text{sim}(d_1, d_2) = \frac{\sum_{(x,y)\in|C_1\cap C_2|} T(t_{1,x} - t_{1,y}, t_{2,x} - t_{2,y})}{|C_1 \cap C_2|}, \tag{7.5}$$

where C_i is as defined in Eq. 7.4, and $t_{i,x}$ is the time, i.e., term number, that course x was taken in d_i, e.g., the first term is numbered 1, the second is numbered 2 and so forth. Note that since students can enroll in summer terms at our university (for one or two courses), we assign the term number for a summer term to half the value of the previous and following spring and fall terms, respectively. This

is to ensure that students who enroll in any summer term have the same term numbers (relative to their entry term) as those who do not enroll in it. Function $T(dt_1, dt_2)$ is defined as:

$$T(dt_1, dt_2) = \begin{cases} 1, & \text{if } dt_1 = dt_2 = 0 \\ \exp\Big(-\lambda(|dt_1 - dt_2|)\Big), & \text{if } dt_1 \times dt_2 \geq 1 \\ 0, & \text{otherwise.} \end{cases} \tag{7.6}$$

where λ is an exponential decay constant[2]. Function T assigns a value of 1 for pairs of courses taken concurrently, i.e., during the same term, in both plans, and assigns a value of 0 for pairs of courses that are either: (i) taken in reversed order in both plans, or (ii) taken concurrently in one plan and sequentially in the other. For pairs of courses taken in the same order, it assigns a positive value that decays exponentially with $|dt_1 - dt_2|$. Our underlying assumption behind such an approach is that, when courses x and y are taken concurrently or in the same order with similar time difference in both d_1 and d_2, then we assume that this is a more similar ordering of both courses than when there is a larger time difference in both plans, and that a different ordering of x and y in the plans does not contribute to their similarity score.

Note that for all the above three pairwise degree similarity metrics, since the degree requirements and courses change from year to year at our university, we only consider pairs of students of the same cohort, i.e., those who entered college in the same term, when computing these metrics. Moreover, we computed each of the Student-to-student Course Time Difference and sequence similarity metrics for pairs of students who have taken at least 20% of their courses in common. We computed these metrics only for the majors where the number of each group of student pairs is ≥ 50 pairs.

7.3 Results

We present the results of our analysis for different groups of students, based on their graduation GPA and TTD. Since both variables are considered important for academic

[2]In our analysis, we chose a small exponential decay constant $\lambda = \frac{1}{5}$ for a slow decay effect.

Table 7.2: Summary of the course timing metrics results among high and low GPA- and TTD-based student groups across all majors.

Metric	Mean	Std.	Mean	Std	Count(†)
	Low TTD		High TTD		Low vs High
Student-to-course Absolute Level Difference	0.293	0.040	0.266	0.080	9 (25)
Student-to-course Signed Level Difference	0.125	0.094	-0.136	0.101	24 (25)
	High GPA		Low GPA		High vs Low
Student-to-course Absolute Level Difference	0.275	0.035	0.331	0.068	13 (25)
Student-to-course Signed Level Difference	0.080	0.072	0.122	0.095	10 (25)

Low and high TTD denote the set of students with time-to-degree that are ≤ 9 and ≥ 11 terms, respectively, both with GPA ≥ 3.0. High and low GPAs denote the set of students with GPAs that are ≥ 3.2 and ≤ 2.8, respectively, both with TTD ≤ 10 terms. The columns "Mean" and "Std." denote the average and standard deviation of the per-major results of the corresponding student group. Count(†) denotes the number of majors that have statistically significant results between the two compared groups, using Welch's t-test with a significance level of 0.001, and the number between parentheses denote the total number of majors that qualified for the corresponding metric.

success, we study the effect of changing one variable while fixing the other to a specific range. For instance, we study the effect of the course timing metrics among students who have low and high TTD of ≤ 9 and ≥ 11 terms, respectively, by assuming that they have achieved a high GPA that is ≥ 3.0.

7.3.1 How does the timing of taking courses with respect to the student's academic level relate to their graduation GPA and TTD?

Figure 7.2 shows the box plots of the 25 majors in terms of the course timing metrics (defined in Section 7.2.2) among different GPA and TTD-based student groups. By comparing the different student groups in terms of Student-to-course Absolute Level Difference, we see that there is no significant difference among the low- and high-TTD-based groups (Fig. 7.2 (a)), while for the high- and low-GPA-based groups (Fig. 7.2 (c)), we see that high GPA students have lower Student-to-course Absolute Level Difference than low GPA ones.

By comparing the student groups in terms of Student-to-course Signed Level Difference, we see that, in Fig. 7.2 (b), low TTD students tend to take more courses ahead of time than high TTD students. On the other hand, Fig 7.2 (d) shows that low GPA students tend to take more courses ahead of time than high GPA students.

To see whether there is statistical significance in these results on a per-major basis,

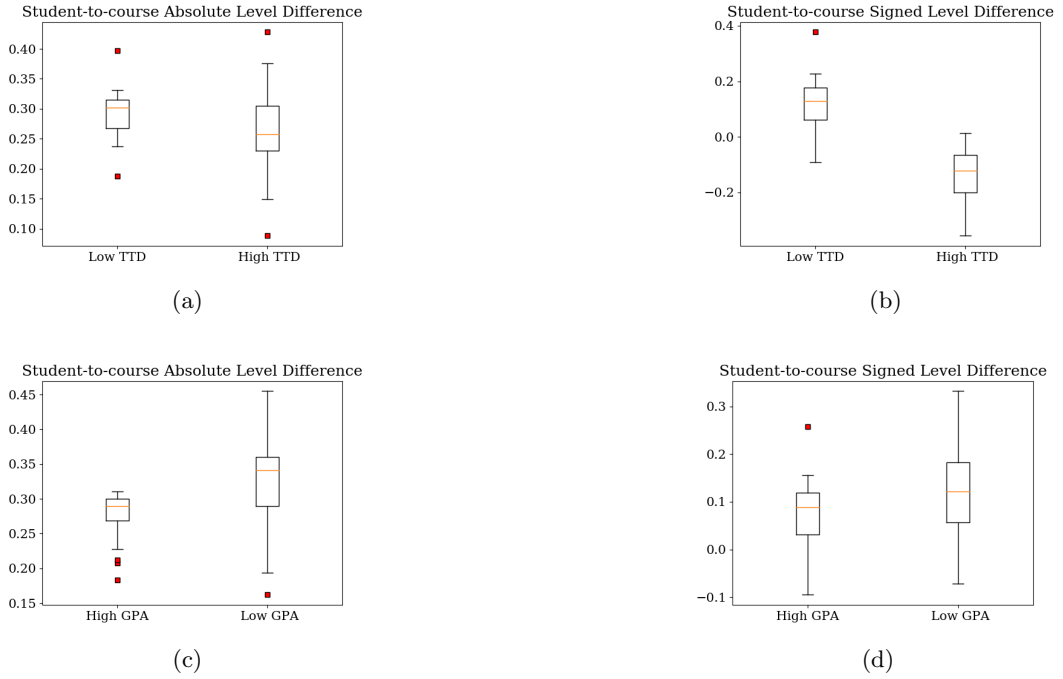


Figure 7.2: Course timing metrics among different groups of full-time students. TTD is shorthand for time-to-degree. Low and high time-to-degree is one that is ≤ 9 and ≥ 11 terms, respectively, both with GPA ≥ 3.0. High and low GPA is one that is ≥ 3.2 and ≤ 2.8, respectively, both with TTD ≤ 10 terms. The line inside the box denotes the median value. The ends of the whiskers denote the lowest datum still within 1.5 IQR (interquartile range) of the lower quartile, and the highest datum still within 1.5 IQR of the upper quartile, while the red squares denote outliers that are outside these ranges.

Table 7.2 shows a summary of the per-major results in terms of the average and standard deviation of the course timing metrics for each student group. It also shows the number of majors that has statistically significant results in one group over the other. These results show that Student-to-course Absolute Level Difference is not a significantly discriminating metric among the different groups of students, as it is statistically significant in 9 and 13 majors only, out of 25 majors, for the TTD- and GPA-based student groups, respectively. In terms of Student-to-course Signed Level Difference, the differences are statistically significant among high and low TTD-based student groups in 24 out of the 25 majors, but only statistically significant in 10 majors among high and low GPA-based groups. This shows that the timing of courses is highly correlated

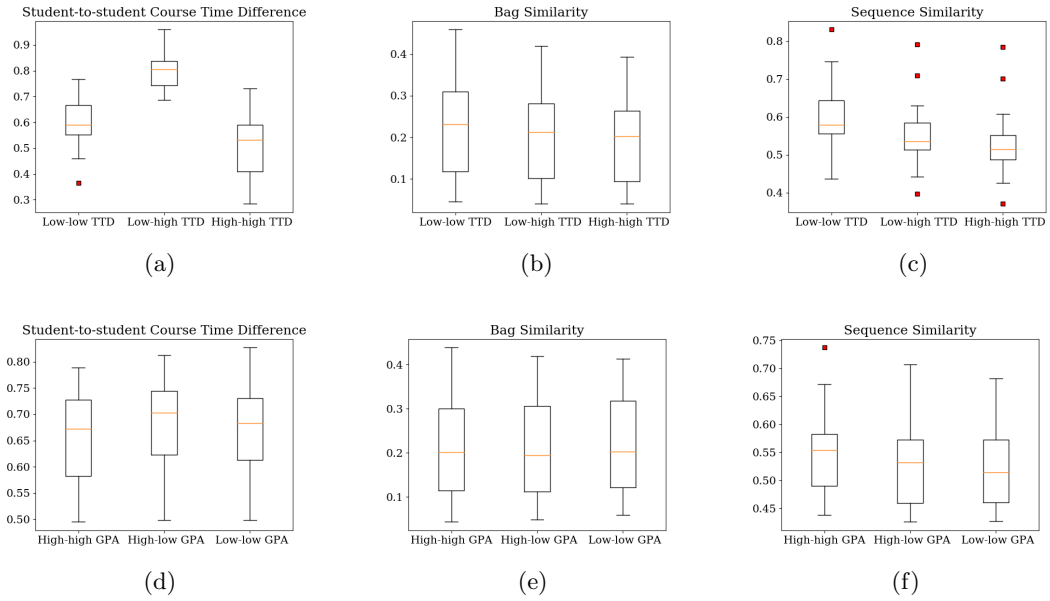


Figure 7.3: Degree similarity metrics among different groups of full-time students. TTD is shorthand for time-to-degree. Low and high time-to-degree is one that is ≤ 9 and ≥ 11 terms, respectively, both with GPA ≥ 3.0. High and low GPA is one that is ≥ 3.2 and ≤ 2.8, respectively, both with TTD ≤ 10 terms. The line inside the box denotes the median value. The ends of the whiskers denote the lowest datum still within 1.5 IQR (interquartile range) of the lower quartile, and the highest datum still within 1.5 IQR of the upper quartile, while the red squares denote outliers that are outside these ranges.

with time to degree, but is not a discriminating factor for the graduation GPA.

7.3.2 How does the pairwise degree similarity between pairs of students relate to the similarity in their graduation GPA and TTD?

Figure 7.3 shows the box plots of the 25 majors in terms of the pairwise degree similarity metrics (defined in Section 7.2.2) among different pairs of GPA and TTD-based student groups, while Table 7.3 shows a summary of the per-major results and the statistical significance between different groups of student pairs. By comparing the different student groups in terms of Student-to-student Course Time Difference in the box plots, we see that for the TTD-based groups (Figs. 7.3 (a)), the high TTD students tend to take their courses at a slightly more similar time together than the low TTD students, with

Table 7.3: Summary of the degree similarity metrics results among different pairs of GPA- and TTD-based student groups across all majors.

Metric	Mean	Std.	Mean	Std.	Mean	Std.	Count(†)		
	LL TTD		LH TTD		HH TTD		LL vs LH	LL vs HH	HH vs LH
CTD	0.595	0.102	0.801	0.081	0.506	0.128	18 (18)	13 (18)	18 (18)
Bag Similarity	0.220	0.124	0.198	0.114	0.190	0.110	25 (25)	23 (25)	16 (25)
Sequence Sim.	0.590	0.099	0.547	0.096	0.528	0.097	18 (18)	14 (18)	14 (18)
	HH GPA		LH GPA		LL GPA		HH vs LH	HH vs LL	LL vs LH
CTD	0.656	0.087	0.677	0.086	0.670	0.088	13 (24)	4 (24)	8 (24)
Bag Similarity	0.209	0.119	0.206	0.115	0.214	0.113	10 (24)	7 (24)	4 (24)
Sequence Sim.	0.550	0.073	0.530	0.070	0.521	0.065	20 (24)	14 (24)	13 (24)

CTD is shorthand for Student-to-student Course Time Difference. LL, LH and HH denote the pairs of students where each pair belongs to the (low, low), (low, high) and (high, high) corresponding student groups, respectively. Low and high TTD denote the set of students with time-to-degree that are ≤ 9 and ≥ 11 terms, respectively, both with GPA ≥ 3.0. High and low GPAs denote the set of students with GPAs that are ≥ 3.2 and ≤ 2.8, respectively, both with TTD ≤ 10 terms. The columns "Mean" and "Std." denote the average and standard deviation of the per-major results of the corresponding student-pair group. Count(†) denotes the number of majors that have statistical significant results between the two compared groups, using Welch's t-test with a significance level of 0.001, and the number between parentheses denote the total number of majors that qualified for the corresponding metric (see Section 7.2.2).

median Student-to-student Course Time Difference value of 0.59 an 0.53, respectively. In addition, pairs of low-high TTD tend to take courses at even a more different timing (with a median Student-to-student Course Time Difference value of 0.8) than pairs of low-low and high-high TTD, aligning with their results of the Student-to-course Signed Level Difference metric in Section 7.3.1. The statistical significance results in Table 7.3 also confirm these differences among low and high TTD-based groups, where 13 out of the 18 qualifying majors have statistically significant Student-to-student Course Time Difference in the pairs of low-low vs pairs of high-high TTD-based groups, while the Student-to-student Course Time Difference is statistically significant in all the 18 majors in each of the pairs of low-low and high-high TTD-based groups vs the pairs of low-high TTD-based groups.

On the other hand, there is not a clear distinction between high and low GPA students in their timing of taking courses among different majors (Fig. 7.3 (d)). Table 7.3 also shows that the average Student-to-student Course Time Difference falls in the range $[0.656, 0.677]$ with a standard deviation of ~ 0.087 among the different GPA-based pairs of students, which also aligns with the results of the course timing metrics in Section 7.3.1 that shows that the timing of courses is not discriminative among different GPA-based student groups.

Table 7.4: Summary of the sequence similarity results among different pairs of GPA- and TTD-based student groups, grouped by their academic division, across all majors.

Division Group	Mean	Std.	Mean	Std.	Mean	Std.	Count(†)		
	LL TTD		LH TTD		HH TTD		LL vs LH	LL vs HH	HH vs LH
Lower Division	0.918	0.024	0.897	0.023	0.896	0.023	17 (20)	11 (20)	8 (20)
Upper Division	0.893	0.024	0.837	0.029	0.806	0.035	25 (25)	23 (25)	23 (25)
	HH GPA		LH GPA		LL GPA		HH vs LH	HH vs LL	LL vs LH
Lower Division	0.916	0.019	0.901	0.026	0.893	0.032	24 (25)	22 (25)	18 (25)
Upper Division	0.881	0.024	0.874	0.023	0.869	0.019	17 (25)	18 (25)	15 (25)

Refer to Section 7.3.2 for the definition of division group. LL, LH and HH denote the pairs of students where each pair belongs to the (low, low), (low, high) and (high, high) corresponding student groups, respectively. Low and high TTD denote the set of students with time-to-degree that are ≤ 9 and ≥ 11 terms, respectively, both with GPA ≥ 3.0. High and low GPAs denote the set of students with GPAs that are ≥ 3.2 and ≤ 2.8, respectively, both with TTD ≤ 10 terms. The columns "Mean" and "Std." denote the average and standard deviation of the per-major results of the corresponding student-pair group. Count(†) denotes the number of majors that have statistical significant results between the two compared groups, using Welch's t-test with a significance level of 0.001, and the number between parentheses denote the total number of majors that qualified for the corresponding metric (see Section 7.2.2).

By comparing the bag similarity among different TTD-based students, we see that low TTD students take more courses in common than high TTD students (average values among pairs of low-low and high-high TTD-based students of 0.22 and 0.19, respectively), with a statistically significant difference in 23 out of the 25 majors.

By looking at the sequence similarity, we see that among the different TTD-based students, low TTD students follow more similar ordering of the courses than high TTD students, with a statistically significant difference (p-value < 0.001) between the two groups in 14 out of the 18 qualifying majors, with an overall sequence similarity that is 0.062 higher in the former group across all majors. An interesting observation is that there is a larger diversity in the sequencing of courses taken by pairs of high TTD students (an average sequence similarity of 0.528) than among pairs of high-low TTD students (an average sequence similarity of 0.547). Along with the course timing results that showed that high TTD students tend to take courses later in time than low TTD students, this could be explained as the former group of students, though they achieve high grades in the courses they take, do not have enough information about their degree requirements. As a result, they end up fulfilling these requirements later in their study than when they should have been fulfilled.

Among different GPA-based students, high GPA students follow more similar sequencing of the courses than low GPA ones, with a difference in their sequence similarities that is statistically significant in 14 out of the 24 qualifying majors (see Section 7.2.2).

To further analyze the differences in the sequence similarity among students, we computed the sequence similarity among different student-pair groups based on their academic levels when they took their courses. At the University of Minnesota, the student is classified into one of four academic levels, based on the total number of credits completed: freshman ($<$ 30 credits), sophomore ($<$ 60 credits), junior ($<$ 90 credits), and senior ($\geq$ 90 credits). Freshmen and sophomores are classified as lower division students, while juniors and seniors are classified as upper division students. Table 7.4 shows the summary of these results, for different pairs of GPA- and TTD-based students. By comparing lower and upper division TTD-based students, we see that there is much greater difference in the different groups' similarities that belong to the upper division than those that belong to the lower division. This shows that students in their early years tend to take courses in a very similar ordering, regardless of their TTD (average sequence similarities of 0.918, 0.897 and 0.896 for low-low, low-high and high-high TTD-based pairs of students, respectively). In their later years, however, low TTD students continue to follow similar sequencing of their courses (with an average sequence similarity of 0.893), while high TTD students diverge from that sequencing and follow more diverse sequencing of their courses (with an average sequence similarity of 0.806).

Similar trends apply to the lower and upper division GPA-based student groups (Table 7.4), though the differences between the sequence similarities of the upper division groups are not as high (average sequence similarities of 0.881, 0.874 and 0.869 for high-high, high-low and low-low GPA-based pairs of students, respectively). This again shows that the sequence similarity is more discriminating for TTD than for GPA.

7.4 Case Study: TTD Prediction

So far, we have analyzed the differences between different GPA- and TTD-based students with respect to the course timing and degree similarity metrics that we defined in

Section 7.2.2. Here, we test whether the timing and ordering of courses as taken by the student at each semester can help predict whether he/she will graduate on-time or over-time. There has been a lot of research on TTD prediction and analyzing the possible effects behind over-time graduation [56–60]. Features like academic features, financial aid, off- and on-campus work and experience, family background, student's demographic information and high school grades have all been investigated and they were found to be good predictors for TTD. In this work, we build a classification model that uses course timing and ordering features to predict students who are at-risk of graduating over-time. We define a student to be at-risk of graduating over-time if he/she graduates in more than four years, i.e., more than nine Fall or Spring terms. We use academic features that have been previously used for TTD prediction as baseline features, to compare their performance against the newly proposed features.

7.4.1 Features

Academic (Baseline) Features

Similar to previous work [56], we use the following academic features that exist in our dataset:

1. **General Experience:** We use the following features: initial status (new vs transfer student), number of program major changes, stop-out time since first enrollment, and number of summer enrollment terms.

2. **Course Grades:** We use percentage of D or F grades, percentage of I (incomplete) or W (withdrawal) grades, individual course grades, and number of repeated courses.

3. **Credit Hours:** We use the total credits accumulated, total transfer credits, percentage of earned to attempted credits, and average credit load per enrolled term.

Course Timing and Ordering (New) Features

Based on the metrics defined in Section 7.2.2 that consider course timing and pairwise course ordering, we define the following features:

1. **Course Timing:** For each course, we use the relative term number when the course is taken (starting from 1), and the academic level of the student when he/she took that course.

2. **Pairwise Course Ordering:** For each pair of courses (c_1, c_2), we use the number of earned credits as well as the number of terms taken between the two terms when the student took c_1 and c_2. Note that a feature "c_1 : earned-credits : c_2" denotes the number of credits that the student earned after taking c_1 and before taking c_2, which is different from the feature "c_2 : earned-credits : c_1", and the same applies for the term difference based features.

7.4.2 Experimental Setup and Evaluation

We normalized each feature to L2 norm as a pre-processing step. We tested many classifiers (including logistic regression, SVM, Decision Tree, Random Forest, and Multi-layer Perceptrons (MLP)) using scikit-learn library in Python [61], and found MLP to be the best performing classifier. The data for each major was trained separately, with an average percentage of over-time graduating students of 54% with a standard deviation of 17%. We constructed different sets of the data, in order to predict whether the student, at each semester, could be at-risk of graduating over-time. We performed 10-fold cross-validation and we report the average results over the 10 folds averaged over all the 25 majors.

We evaluate the classifier's performance in terms of the following metrics:

- Recall of at-risk: Recall is the ratio of true positives to all actual positives.

- Precision of at-risk: Precision is the ratio of true positives to all predicted positives.

- F_1 of at-risk: F_1 score is the harmonic mean between Precision and Recall, which conveys the balance between the two and computed as:

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{7.7}$$

- Area under the receiver operating characteristic (ROC) curve (AUC): ROC curve plots the true positive rate against the false positive rate, at various thresholds.

AUC corresponds to the probability that the classifier will rank a random positive instance higher than a negative one.

Table 7.5: TTD prediction results using the academic (baseline) and new (course timing and ordering) features.

Metric	Feature Type	Semester Number					
		2	3	4	5	6	7
F1 of at-risk	Academic	0.420	0.424	0.445	0.405	0.435	0.417
	Course Timing and Ordering	0.388	0.419	0.407	0.424	0.463	0.418
Precision of at-risk	Academic	0.365	0.369	0.387	0.352	0.379	0.367
	Course Timing and Ordering	0.363	0.385	0.366	0.389	0.424	0.390
Recall of at-risk	Academic	0.528	0.542	0.557	0.514	0.551	0.525
	Course Timing and Ordering	0.478	0.522	0.519	0.546	0.575	0.520
AUC	Academic	0.550	0.549	0.548	0.548	0.547	0.548
	Course Timing and Ordering	0.549	0.545	0.544	0.546	0.545	0.544

Underlined entries denote the best performance across the two feature types in each semester.

7.4.3 Experimental Results

Table 7.5 shows the TTD prediction results when using the academic and course timing and ordering features, by predicting TTD at each semester when the student is enrolled, starting from the second to the seventh semester. The results show that the prediction performance using the proposed course timing and ordering features is similar to that using the standard academic features. Using the course timing and ordering features tends to give more accurate F_1, precision and recall scores in the late years (semesters 5 though 7) than in early years (semesters 2 through 4). In terms of AUC, there are small insignificant differences in the prediction performance using both types of features. This shows that degree planning in terms of the timing of courses and the ordering between them plays an important role in the student's TTD, that is similarly equal to his/her general experience and academic performance in terms of grades and credit hours.

7.5 Summary

In this chapter, we investigated how the student's academic level when they take different courses as well as the pairwise degree similarity between pairs of students relate to their graduation GPA and time to degree (TTD). Our analyses were conducted on a

large-scale dataset that spans 16 years worth of degree plans pursued by students from 25 majors from different colleges at the University of Minnesota. Our findings indicated that:

- Student clusters that are based on their graduation GPA or TTD tend to share more similarities within themselves than with students from different clusters, in the time when they take their courses as well as the set and sequencing of them.
- Low TTD students tend to take courses ahead of time. In addition, they follow more similar sequencing for the common courses, especially in their late years than high TTD students.
- Low GPA students tend to take courses ahead of time and follow more diverse sequencing of their courses than high GPA students.

Overall, there is a strong correlation between the timing and ordering of courses and the students' TTD. However, the correlation between them and the student's graduation GPA is not as strong. One potential explanation for this could be that, since each course provides a specific set of knowledge components that can be useful or required for other courses, there is an inherent sequencing among courses through which the students can accumulate their knowledge in a correct way and graduate on time. However, even when students follow the correct sequencing that guarantees on-time graduation, their grades in different courses can be affected by many other factors that can or cannot be measured. For instance, the student's effort in the course and how much time they allocate for learning its material and finishing its assignments and projects is hard to measure in the actual classroom setting. Another factor could be the student's learning style and how it aligns with the instructor's teaching style, the types of evaluation they do, as well as the grading system they follow. A third factor could be the student's network in class and whether they have a good support for understanding the material inside and outside of class. All these factors play an important role in the student's performance in class and hence affect their final grades that together make up their graduation GPA.

From a research perspective, this study contributes to the literature by providing empirical evidence about the timing and ordering of courses as pursued by past students

and how these relate to their graduation GPA and TTD. Researchers who develop data-driven approaches that make use of past degree plans, such as course recommendation, course sequencing, and curriculum designing, can use this information to better model the degree plans and develop more robust methods that can better assist students towards academic success, by graduating on-time with high GPA.

From an advisor perspective, this study makes a step forward towards understanding the importance of the timing and ordering of courses and how they are related to the student's graduation GPA and TTD. Advisors can use this information to better guide their students to take courses in the right time that can help them towards their academic success. They can also help them designing their own personalized plans and modify them based on their current performance and end goals, as well as show them the trade-offs they might have to make with respect to their expected graduation GPA and TTD.

From a learner perspective, knowing how the timing and sequencing of courses is related to their academic performance, especially their TTD, students can have better knowledge about how to plan their degree in order to graduate on time and save more money by taking the right set of courses in the optimal sequencing that will help them towards successful graduation in a timely manner.

Since the analysis was conducted on a large-scale dataset that spans 16 years and contains 25 majors from different colleges, we believe that the results of this analysis are generalizable and can apply on data from other universities.

There are some limitations to the current study that readers need to keep in mind for future research. Firstly, this study does not study the effect of the timing of taking courses on the students' grades in these individual courses, i.e., whether taking a course at the same, higher or lower level than the student's academic level will be related to the student's grade in this course. If such a correlation exists, then data-driven approaches need to take this into account while utilizing the degree plans. Secondly, this study does not analyze the causal inference between ordering and timing of courses and academic performance, to test whether one leads to the other. Lastly, we did not study the competition and synergy among courses taken in the same term. This might also affect the student's academic performance, since students have limited amount of time to study for the courses they take simultaneously, which creates competition among these

courses. On the other hand, there might be courses in which the knowledge that one course provides during the term helps with the understanding of another course, which creates synergy among them. We plan to address these limitations in the future.

Our study has pointed out some good insights about the timing and sequencing of courses that both students and their advisors could pay attention to. However, further analysis and qualitative research is needed to identify other factors that might affect these results, such as the dynamics of the whole network of students and if closer fellows tend to take more courses together and how this affects their grades. Nonetheless, this study points towards the need for the data-driven approaches that work on course recommendation and sequencing or curriculum designing to consider the differences in the degree plans and know how to best utilize them.

Chapter 8

Grade-aware Course Recommendation Approaches

Both *course recommendation* [16, 17, 22, 53, 54] and *grade prediction* [4, 5, 16, 62, 63] methods aim to help students during the process of course registration in each semester. By learning from historical registration data, course recommendation focuses on recommending courses to students that will help them in completing their degrees. Grade prediction focuses on estimating the students' expected grades in future courses. Based on what courses they previously took and how well they performed in them, the predicted grades give an estimation of how well students are prepared for future courses. Nearly all of the previous studies have focused on solving each problem separately, though both problems are inter-related in the sense that they both aim to help students graduate in a timely and successful manner.

In this chapter, we propose a *grade-aware course recommendation* framework that focuses on recommending a set of courses that will help students: (i) complete their degrees in a timely fashion, and (ii) maintain or improve their GPA. To this end, we propose two different approaches for recommendation. The first approach ranks the courses by using an objective function that differentiates between courses that are expected to increase or decrease a student's GPA. The second approach uses the grades that students are expected to obtain in future courses to improve the ranking of the courses produced by course recommendation methods.

To obtain course rankings in the first approach, we adapt two widely-known representation learning techniques, which have proven successful in many fields, to solve the grade-aware course recommendation problem. The first is based on Singular Value Decomposition (SVD), which is a linear model that learns a low-rank approximation of a given matrix. The second, which we refer to as Course2vec, is based on Word2vec [33] that uses a log-linear model to formulate the problem as a maximum likelihood estimation problem. In both approaches, the courses taken by each student are treated as temporally-ordered sets of courses, and each approach is trained to learn these orderings.

In the following sections, we present the summary of our results in Sec. 8.1. Then, we explain our methods in Sec. 8.2, describe the experimental setup and evaluation methodology in Sec. 8.3, discuss the results in Sec. 8.4, Sec. 8.5 and Sec. 8.6, and summarize the Chapter in Sec. 8.7.

8.1 Main Contributions

Our contributions in this chapter are the following:

1. We propose a *Grade-aware Course Recommendation* framework in higher education that recommends courses to students that the students are most likely to register for in their following terms and that will help maintain or improve their overall GPA. The proposed framework combines the benefits of both course recommendation and grade prediction approaches to better help students graduate in a timely and successful manner.

2. We investigate two different approaches for solving grade-aware course recommendation. The first approach uses an objective function that explicitly differentiates between good and bad courses, while the other approach combines grade prediction methods with course recommendation methods in a non-linear way.

3. We adapt two-widely used representation learning techniques to solve the grade-aware course recommendation problem, by modeling historical course ordering data and differentiating between courses that increase or decrease the student's GPA.

4. We performed an extensive set of experiments on a dataset spanning 16 years obtained from the University of Minnesota, which includes students who belong to 23 different majors. The results show that: (i) the proposed grade-aware course recommendation approaches outperform grade-unaware course recommendation methods in recommending more courses that increase the students' GPA and fewer courses that decrease it; and (ii) the proposed representation learning approaches outperform competing approaches for grade-aware course recommendation in terms of recommending courses which students are expected to perform well in, as well as differentiating between courses which students are expected to perform well in and those which they are expected not to perform well in.

8.2 Proposed Approaches

Undergraduate students often achieve inconsistent grades in the various courses they take, which may increase or decrease their overall GPA. This is illustrated in Figure 8.1 that shows the histogram of differences between each grade obtained by a student over his/her prior average grade, for the dataset used in our experiments (Table 8.1). As we can see, more than 10% of the grades are a full-letter grade lower, than the corresponding students' previous average grades[1]. The poor performance in some of these courses can result in students having to retake the same courses at a later time, or increase the number of courses that they will have to take in order to graduate with a desired GPA. As a result, this will increase the financial cost associated with obtaining a degree and can incur an opportunity cost by delaying the students' graduation.

For the cases in which a student's performance in a course is a result of him/her not being well-prepared for it (i.e., is taking the course at the wrong time in his/her studies), course recommendation methods can be used to recommend a set of courses for that student that will help: (i) him/her in completing his/her degree in a timely fashion, and (ii) maintain or improve his/her GPA. We will refer to the methods that do those simultaneously as **grade-aware course recommendation** approaches. Note that the majority of the existing approaches cannot be used to solve this problem as

[1]The letter grading system in this dataset has 11 letter grades (A, A-, B+, B, B-, C+, C, C-, D+, D, F) that correspond to the numerical grades (4, 3.67, 3.33, 3, 2.67, 2.33, 2, 1.67, 1.33, 1, 0), with A being the highest grade and F the lowest one.

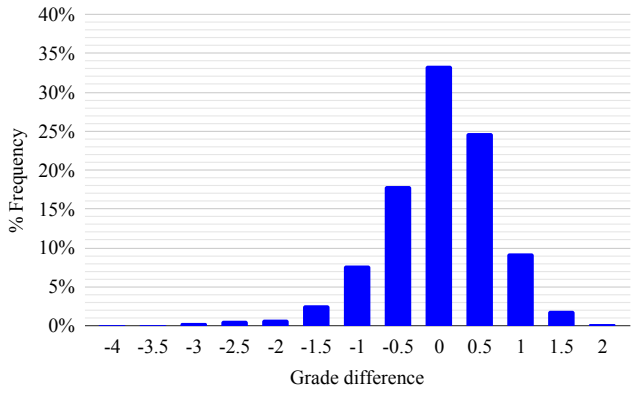


Figure 8.1: Grade difference from the student's average previous grade.

they ignore the performance the student is expected to get in the courses that they recommend.

In this work, we propose two different approaches for grade-aware course recommendation. The first approach (Sec. 8.2.1) uses two representation learning approaches that explicitly differentiate between courses in which the student is expected to perform well in and courses in which the student is expected not to perform well in. The second approach (Sec. 8.2.2) combines grade prediction methods with course recommendation methods to improve the final course rankings. The goal of both approaches is to rank the courses in which the student is expected to perform well in higher than those in which he/she is expected not to perform well in.

8.2.1 Grade-aware Representation Learning Approaches

Motivated by our findings in Chapter 7, where we saw that students follow different sequencing for their courses and that this sequencing is related to their graduation GPA, our first approach for solving the grade-aware course recommendation problem relies on modeling the sequencing of courses and differentiating between courses which the student is expected to perform well in and courses which the student is expected not to perform well in. As such, for every student, we define a course taken by him/her to be **a good (subsequent) course** if the student's grade in it is equal to or higher than his/her average previous grade, otherwise, we define that course to

be **a bad (subsequent) course**. The goal of our method is to recommend to each student a set of good courses.

Motivated by the success of representation learning approaches in recommendation systems [29, 38–40], we adapt two widely-used approaches to solve the grade-aware course recommendation problem. The first approach applies Singular Value Decomposition linear factorization model on a co-occurrence frequency matrix that differentiates between good and bad courses (Sec. 8.2.1), while the second one optimizes an objective function of a neural network log-linear model that differentiates between good and bad courses (Sec. 8.2.1).

In both approaches, the courses taken by each student are treated as temporally-ordered sets of courses, and each approach is trained on this data in order to learn the proper ordering of courses as taken by students. The course representations learned by these models are then used to create personalized rankings of courses for students that are designed to include courses that are relevant to the students' degree programs and will help them maintain or increase their GPAs.

Singular Value Decomposition (SVD)

SVD [25] is a traditional low-rank linear model that has been used in many fields. It factorizes a given matrix $\mathbf{X}$ by finding a solution to $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where the columns of $\mathbf{U}$ and $\mathbf{V}$ are the left and right singular vectors, respectively, and $\mathbf{\Sigma}$ is a diagonal matrix containing the singular values of $\mathbf{X}$. The d largest singular values, and corresponding singular vectors from $\mathbf{U}$ and $\mathbf{V}$, is the rank d approximation of $\mathbf{X}$ ($\mathbf{X}_d = \mathbf{U}_d\mathbf{\Sigma}_d\mathbf{V}_d^T$). This technique is called truncated SVD.

Since we are interested in learning course ordering as taken by past students, we apply SVD on a previous-subsequent co-occurrence frequency matrix $\mathbf{F}$, where F_{ij} is the number of students in the training data that have taken course i before they took course j.

We form two different previous-subsequent co-occurrence frequency matrices, as follows. Let n_{ij}^+ and n_{ij}^- be the number of students who have taken course i before course j, where course j is considered a good course for the first group and a bad course for the second one, respectively. The two matrices are:

1. $\mathbf{F}^+$: where $F_{ij}^+ = n_{ij}^+$.

2. $\mathbf{F}^{+-}$: where $F_{ij}^{+-} = n_{ij}^{+} - n_{ij}^{-}$.

We scaled the rows of each matrix to $L1$ norm and then applied truncated SVD on them. The course embeddings are then given by $\mathbf{U}_d\sqrt{\mathbf{\Sigma}_d}$ and $\mathbf{V}_d\sqrt{\mathbf{\Sigma}_d}$ for the previous and subsequent courses, respectively.

Note that we append a (+), or (+-) as a superscript to the matrix and as a suffix to the corresponding method's name based on what course information it utilizes during learning and how it utilizes it. A (+)-based method utilizes the good course information only and ignores the bad ones, while a (+-)-based method utilizes both the good and bad course information and differentiates between them.

Recommendation. Given the previous and subsequent course embeddings estimated by SVD, course recommendation is done as follows. Given a student s with his/her previously-taken set of courses, $c_1, \ldots, c_k$, who would like to register for his/her following term, we compute his/her implicit profile by averaging over the embeddings of the courses taken by him/her in all previous terms[2]. We then compute the dot product between s's profile and the embeddings of each candidate course $c_t \in C$. Then, we rank the courses in non-increasing order according to these dot products, and select the top courses as the final recommendations for s.

Course2vec

The above SVD model works on pairwise, one-to-one relationships between previous and subsequent courses. We also model course ordering using a many-to-one, log-linear model, which is motivated by the recent word2vec Continuous Bag-Of-Word (CBOW) model [33]. Word2vec works on sequences of individual words in a given text, where a set of nearby (context) words (i.e., words within a pre-defined window size) are used to predict the target word. In our case, the sequences would be the ordered terms taken by each student, where each term contains a set of courses, and the previous set of courses would be used to predict future courses for each student.

[2]Note that we tried using different window sizes for the number of previous terms. Using all previous terms achieved the best results than using one, two or three previous terms only.

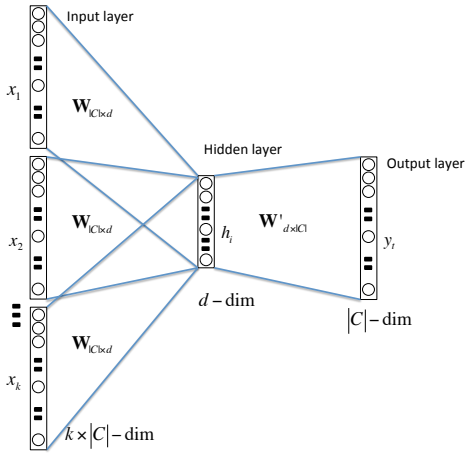


Figure 8.2: Neural network architecture for Course2vec.

Model Architecture. We formulate the problem as a maximum likelihood estimation problem. Let $\mathcal{T}_i = \{c_1, \ldots, c_n\}$ be a set of courses taken in some term i. A sequence $Q_s = \langle \mathcal{T}_1, \ldots, \mathcal{T}_m \rangle$ is an ordered list of m terms as taken by some student s, where each term can contain one or more courses. Let $\mathbf{W} \in \mathbb{R}^{|C|\times d}$ be the courses' representations when they are treated as *previous* courses, and let $\mathbf{W'} \in \mathbb{R}^{d\times|C|}$ be their representations when they are treated as "subsequent" courses, where $|C|$ is the number of courses and d is the number of dimensions in the embedding space. We define the probability of observing a future course c_t given a set of previously-taken courses $c_1, \ldots, c_k$ using the softmax function, i.e.,

$$Pr(c_t|c_1, \ldots, c_k) = y_t = \frac{\exp(\mathbf{w'}_{c_t}^T \mathbf{h})}{\sum_{j=1}^{C} \exp(\mathbf{w'}_{c_j}^T \mathbf{h})}, \tag{8.1}$$

where $\mathbf{h}$ denotes the aggregated vector of the representations of the previous courses, where we use the average pooling for aggregation, i.e.,

$$\mathbf{h} = \frac{1}{k}\mathbf{W}^T(\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_k),$$

where $\mathbf{x}_i$ is a one-hot encoded vector of size $|C|$ that has 1 in the c_i's position and 0 otherwise. The Architecture for Course2vec is shown in Figure 8.2. Note that one may consider more complex neural network architectures, which is left for future work.

We propose the two following models:

1. **Course2vec(+)**. This model maximizes the log-likelihood of observing only the good subsequent courses that are taken by student s in some term given his/her previously-taken set of courses. The objective function of Course2vec(+) is thus:

$$\underset{\mathbf{W},\mathbf{W}'}{\text{maximize}} \sum_{s\in\mathcal{S}} \sum_{\mathcal{T}_i\in Q_s} \Big(\log Pr(\mathcal{G}_{s,i}|\mathcal{P}_{s,i}) \Big), \tag{8.2}$$

   where: $\mathcal{S}$ is the set of students, $\mathcal{G}_{s,i}$ is the set of good courses taken by student s at term i, and $\mathcal{P}_{s,i}$ is the set of courses taken by student s prior to term i. Note that i starts from 2, since the previous set of courses $\mathcal{P}_{s,i}$ would be empty for $i = 1$.

2. **Course2vec(+-)**. This model maximizes the log-likelihood of observing good courses and minimizes the log-likelihood of observing bad courses given the set of previously-taken courses. The objective function of Course2vec(+-) is thus:

$$\underset{\mathbf{W},\mathbf{W}'}{\text{maximize}} \sum_{s\in\mathcal{S}} \sum_{\mathcal{T}_i\in Q_s} \Big(\begin{array}{l} \log Pr(\mathcal{G}_{s,i}|\mathcal{P}_{s,i}) \\ \qquad - \log Pr(\mathcal{B}_{s,i}|\mathcal{P}_{s,i}) \Big), \end{array} \tag{8.3}$$

   where: $\mathcal{B}_{s,i}$ is the set of bad courses taken by student s at term i, and the rest of the terms are as defined in Eq. 8.2.

Note that Course2vec(+) is analogous to SVD(+) and Course2vec(+-) is analogous to SVD(+-) in terms of how they utilize the good and bad courses in the training set.

Model Optimization. The objective functions in Eqs. 8.2 and 8.3 can be solved using Stochastic Gradient Descent (SGD), by solving for one subsequent course at a time. The computation of gradients in the two equations requires computing Eq. 8.1 for all courses for the denominator, which requires knowing whether a course is to be considered a good or a bad subsequent course for a given context. However, not all the relationships between every context (previous set of courses) and every subsequent course is known from the data. Hence, for each context, we only update the subsequent course vector when the course is known to be a good or bad subsequent course associated with that context. In the case that some context does not have a sufficient pre-defined

number of subsequent courses with known relationships, then we randomly sample a few other courses and treat them as bad courses, similar to the negative sampling approach used in word2vec [34].

Note that in Course2vec(+-), since a course can be seen as both a good and a bad subsequent course for the same context in the data (for different students), then, in this case, we randomly choose whether to treat that course as good or bad each time according to a uniform distribution that is based on its good and bad frequency in the dataset. In addition, for both Course2vec(+) and Course2vec(+-), if the frequency between a context and a subsequent course is less than a pre-defined threshold, e.g., 20, then we randomly choose whether to update that subsequent course's vector in the denominator each time it is visited. The code for Course2vec can be found at: `https://goo.gl/uCCqie`, which is built on the original word2vec code that was implemented for the CBOW model[3].

Recommendation Given the previous and subsequent course embeddings estimated by Course2vec, course recommendation is done as follows. Given a student s with his/her previously-taken set of courses, $c_1, \ldots, c_k$, who would like to register for his/her following term, we compute the probability $Pr(c_t|c_1, \ldots, c_k)$ for each candidate course $c_t \in C$ according to Eq. 8.1. We then rank the courses in non-increasing order according to their probabilities, and select the top courses as the final recommendations for s. Note that since the denominator in Eq. 8.1 is the same for all candidate courses, the ranking score for course c_t can be simplified to the dot product between $\mathbf{w'}_{c_t}$ and $\mathbf{h}$, where $\mathbf{h}$ represents the student's implicit profile.

8.2.2 Combining Course Recommendation with Grade Prediction

The second approach that we developed for solving the grade-aware course recommendation problem relies on using the grades that students are expected to obtain in future courses to improve the ranking of the courses produced by course recommendation methods. Our underlying hypothesis behind this approach is that, a course that both is ranked high by a course recommendation method and has a high predicted grade

[3]Original code is at: `https://goo.gl/UvUuMQ`

should be ranked higher than one that either has a lower ranking by the recommendation method or is predicted to have a lower grade in it. This in turn will help improve the final course rankings for students by taking both scores into account simultaneously.

Let $\hat{g}_{s,c}$ be the predicted grade for course c as generated from some grade prediction model, and let $\hat{r}_{s,c}$ be the ranking score for c as generated from some course recommendation method. We combine both scores to compute the final ranking score for c as follows:

$$\text{rank-score}_{s,c} = \hat{g}_{s,c}^{\alpha} \times (|\hat{r}_{s,c}|)^{(1-\alpha)} \times \text{sign}(\hat{r}_{s,c}), \tag{8.4}$$

where α is a hyper-parameter in the range $(0, 1)$ that controls the relative contribution of $\hat{g}_{s,c}$ and $\hat{r}_{s,c}$ to the overall ranking score, and $\text{sign}(\hat{r}_{s,c})$ denotes the sign of $\hat{r}_{s,c}$, i.e., 1 if $\hat{r}_{s,c}$ is positive and -1 otherwise. Note that both $\hat{g}_{s,c}$ and $\hat{r}_{s,c}$ are standardized to have zero mean and unit variance.

In this work, we will use the representation learning approaches described in Sec. 8.2.1 as the course recommendation method. We will also use the grade-unaware variations of each of them (see Sec. 8.3.2) to compare combining the grade prediction methods with both recommendation approaches.

To obtain the grade prediction score, we will use *Cumulative Knowledge-based Regression Models* [62], or CKRM for short. CKRM is a set of grade-prediction methods that learn low-dimensional as well as textual-based representations for courses that denote the required and provided knowledge components for each course. It represents a student's knowledge state as the sum of the provided knowledge component vectors of the courses taken by them, weighted by their grades in them. CKRM then predicts the student's grade in a future course as the dot product between their knowledge state vector and the course's required knowledge component vector. We will denote the recommendation method that combines CKRM with SVD and Course2vec as **CKRM+SVD** and **CKRM+Course2vec**, respectively.

8.3 Experimental Evaluation

8.3.1 Dataset Description and Preprocessing

The data used in our experiments was obtained from the University of Minnesota, where it spans a period of 16 years (Fall 2002 to Summer 2017). From that dataset,

we extracted the degree programs that have at least 500 graduated students until Fall 2012, which accounted for 23 different majors from different colleges. For each of these degree programs, we extracted all the students who graduated from this program and extracted the 50 most frequent courses taken by the students as well as the courses that belonged to frequent subjects, e.g., CSCI is a subject that belongs to the Computer Science department at the University. A subject is considered to be frequent if the average number of courses that belong to that subject over all students is at least three. This filtering was made to remove the courses we believe are not relevant to the degree program of students. We also removed any courses that were taken as pass/fail.

Using the above dataset, we split it into train, validation and test sets as follows. All courses taken before Spring 2013 were used for training, courses taken between Spring 2013 and Summer 2014 inclusive were used for validation, and courses taken afterwards (Fall 2014 to Summer 2017 inclusive) were used for test purposes.

At the University of Minnesota, the letter grading system has 11 letter grades (A, A-, B+, B, B-, C+, C, C-, D+, D, F) that correspond to the numerical grades (4, 3.667, 3.333, 3, 2.667, 2.333, 2, 1.667, 1.333, 1, 0). For each (context, subsequent) pair in the training, validation, and test set, where the context represents the previously-taken set of courses by a student, the context contained only the courses taken by the student with grades higher than the D+ letter grade. The statistics of the 23 degree programs are shown in Table 8.1.

8.3.2 Baseline and Competing Methods

We compare the performance of the proposed representation learning approaches against competing approaches for grade-aware course recommendation, which are defined as follows:

- **Grp-pop(+-)**: We modify the group popularity ranking method developed in [16] and explained in Sec. 3.2 to solve the grade-aware course recommendation. For each course c, let n_c^+ and n_c^- be the number of students that have the same major and academic level as the target student s, where c was considered a good subsequent course for the first group and a bad one for the second group. We can differentiate between good and bad subsequent courses using the following ranking

Table 8.1: Dataset statistics.

Major	# Students	# Courses	# Grades
Accounting (ACCT)	661	55	7,614
Aerospace Engr. (AEM)	866	72	13,280
Biology (BIOL)	1,927	113	15,590
Biology, Soc. & Envir. (BSE)	1,231	56	9,389
Biomedical Engr. (BME)	1,002	64	13,808
Chemical Engr. (CHEN)	1,045	82	10,219
Chemistry (CHEM)	765	78	7,814
Civil Engr. (CIVE)	1,160	74	15,992
Communication Studies (COMM)	2,547	90	17,135
Computer Science & Engr. (CSE)	1,790	98	13,520
Electrical Engr. (ECE)	1,197	84	12,781
Elementary Education (ELEM)	1,283	60	15,303
English (ENGL)	1,790	113	12,451
Finance (FIN)	1,326	55	12,150
Genetics, Cell Biol. & Devel. (GCD)	843	92	9,726
Journalism (JOUR)	2,043	91	23,549
Kinesiology (KIN)	1,499	161	23,451
Marketing (MKTG)	2,077	51	13,084
Mechanical Engr. (MECH)	1,501	79	25,608
Nursing (NURS)	1,501	88	18,239
Nutrition (NUTR)	940	71	12,400
Political Science (POL)	1,855	111	13,904
Psychology (PSY)	3,047	100	25,299

score (which is similar to the (+-)-based approaches):

$$\text{rank-score}_{s,c} = n_c^+ - n_c^-. \tag{8.5}$$

- **Grp-pop(+)**: Here, the group popularity ranking method considers only the good subsequent courses, similar to SVD(+) and Course2vec(+). Specifically, the ranking score is computed as:

$$\text{rank-score}_c = n_c^+,$$

  where n_c^+ is as defined in Eq. 8.5.

- **Course dependency graph**: This is the course recommendation method utilized in [24] (see Sec. 3.2).

We also compare the performance of the representation learning approaches for both grade-aware and grade-unaware course recommendation. The grade-unaware representation learning approaches are defined as follows:

- **SVD(++)**: Here, SVD is applied on the previous-subsequent co-occurrence frequency matrix: $\mathbf{F}^{++}$: where $F_{ij}^{++} = n_{ij}^{+} + n_{ij}^{-}$.

- **Course2vec(++)**. This model maximizes the log-likelihood of observing all courses taken by student s in some term given the set of previously-taken courses, regardless of the subsequent course being a good or a bad one. This can be written as:

$$\underset{\mathbf{W},\mathbf{W}'}{\text{maximize}} \sum_{s \in \mathcal{S}} \sum_{\mathcal{T}_i \in Q_s} \Big(\log Pr(\mathcal{C}_{s,i} | \mathcal{P}_{s,i}) \Big),$$

  where: $\mathcal{C}_{s,i}$ is the set of courses taken by student s at term i, and the rest of the terms are as defined in Eq. 8.2.

Note that, here we append a (++) suffix to the grade-unaware variation of the method's name since it utilizes all the course information without differentiating between good and bad courses.

8.3.3 Evaluation Methodology and Metrics

Previous course recommendation methods used the recall metric to evaluate the performance of their methods. The goal of the proposed grade-aware course recommendation methods is to recommend to the student courses which he/she is expected to perform well in and not recommend courses which he/she is expected not to perform well in. As a result, we cannot use the recall metric as is, and instead, we use three variations of it that differentiate between good and bad courses. The first, *Recall(good)*, measures the fraction of the actual good courses that are retrieved. The second, *Recall(bad)*, measures the fraction of the actual bad courses that are retrieved. The third, *Recall(diff)*, measures the overall performance of the recommendation method in ranking the good courses higher than the bad ones.

The first two metrics are computed as the average of the student-term-specific corresponding recalls. In particular, for a student s and a target term t, the first two recall metrics for that (s, t) tuple are computed as:

1. $\text{Recall(good)}_{(s,t)} = \frac{\left|G_{s,n_{(s,t)}}\right|}{n^{g}_{(s,t)}}$.

2. Recall(bad)$_{(s,t)} = \frac{\left|B_{s,n_{(s,t)}}\right|}{n^b_{(s,t)}}$.

$G_{s,n_{(s,t)}}$ and $B_{s,n_{(s,t)}}$ denote the set of good and bad courses, respectively, that were taken by s in t and exist in his/her list of $n_{(s,t)}$ recommended courses, $n_{(s,t)}$ is the actual number of courses taken by s in t, and $n^g_{(s,t)}$ and $n^b_{(s,t)}$ are the actual number of good and bad courses taken by s in t, respectively. Since our goal is to recommend good courses only, we consider a method to perform well when it achieves a high Recall(good) and a low Recall(bad).

Recall(diff) is computed as the difference between Recall(good) and Recall(bad), i.e.,

3. Recall(diff) = Recall(good) - Recall(bad).

Recall(diff) is thus a signed measure that assesses both the degree and direction to which a recommendation method is able to rank the actual good courses higher than the bad ones in its recommended list of courses for each student, so the higher the Recall(diff) value, the better the recommendation method is.

To further analyze the differences in the ranking results of the proposed approaches, we also computed the following two metrics:

- **Percentage GPA increase/decrease:** Let cur-good$_s$ and cur-bad$_s$ be the current GPA achieved by student s on the good and bad courses recommended by some recommendation method, respectively, and let prev-gpa$_s$ be his/her GPA prior to that term. Then, the percentage GPA increase and decrease are computed as:

$$\% \text{ GPA increase} = \frac{\text{cur-good}_s - \text{prev-gpa}_s}{\text{prev-gpa}_s} \times 100.0.$$

$$\% \text{ GPA decrease} = \frac{\text{prev-gpa}_s - \text{cur-bad}_s}{\text{prev-gpa}_s} \times 100.0.$$

- **Coverage for good/bad terms:** The number of terms where some recommendation method recommends good (or bad) subsequent courses to will be referred to as its coverage for good (or bad) terms. The higher the coverage for good terms by some method, the more students who will get good recommendations that will maintain or improve their overall GPA. On the other hand, the lower the

coverage for bad terms, the less students who will get bad recommendations that will decrease their overall GPA.

We compute the above two metrics for the terms on which the recommendation method recommends at least one of the actual courses taken in that term. For each method, the percentage GPA increase and decrease as well as the coverage for good and bad terms are computed as the average of the individual scores. Since we would like to recommend courses that optimize the student's GPA, the higher the GPA percentage increase and the coverage for good terms and the lower the GPA percentage decrease and the coverage for bad terms that a method achieves, the better the method is.

Note that, a recommendation is only done for students who have taken at least three previous courses. For each (s, t) tuple, the recommended list of courses using any method are selected from the list of courses that are being offered at term t only, and that were not already taken by s with an associated grade that is either: (i) $\geq$ C+, or, (ii) $\geq \mu_s - 1.0$, where μ_s is the average previous grade achieved by s. Therefore, we only allow recommending repeated courses in the case that the student has achieved a low grade in it such that the course's credits do not add to the earned credits, or when they a achieve bad grade in them relative to their grades in previous terms. This filtering technique significantly improved the performance of all the baseline and proposed methods.

8.3.4 Model Selection

We did an extensive search in the parameter space for model selection. The parameters in the SVD-based models is the number of latent dimensions (d). The parameters in the Course2vec-based models are: the number of latent dimensions (d), and the minimum number of subsequent courses (*samples*), in the denominator of Eq. 8.1 that are used during the SGD process of learning the model. We experimented with the parameter d in the range $[10 - 30]$ with a step of 5, with the minimum number of *samples* with the values $\{3, 5\}$, and with the parameter α in Eq. 8.4 in the range $[0.1 - 0.9]$ with a step of 0.2.

The training set was used for learning the distributed representations of the courses, whereas the validation set was used to select the best performing parameters in terms

of the highest Recall(diff).

8.4 Results

We evaluate the effectiveness of the proposed grade-aware course recommendation methods in order to answer the following questions:

RQ1. How do the SVD- and Course2vec-based approaches for course recommendation compare to each other?

RQ2. How do the combination of grade prediction with representation learning approaches compare to each other?

RQ3. How do the two proposed approaches for solving grade-aware course recommendation compare to each other?

RQ4. How do the proposed approaches compare to competing approaches for grade-aware course recommendation?

RQ5. What are the benefits of grade-aware course recommendation over grade-unaware course recommendation?

RQ6. How does the recommendation accuracy vary across different student sub-groups?

RQ7. What are the characteristics of the courses recommended by our proposed models, in terms of the course difficulty and popularity?

8.4.1 Comparison of the Representation Learning Approaches for Grade-aware Course Recommendation

Table 8.2 shows the prediction performance of the two proposed representation learning approaches for grade-aware course recommendation. SVD(+) achieves the best Recall(good), while SVD(+-) achieves the best Recall(diff). Course2vec(+-) achieves the best Recall(bad), which is comparable to SVD(+-).

By comparing the corresponding SVD and Course2vec approaches, we see that SVD outperforms Course2vec in almost all cases. We believe this is caused by the fact that there is a limited number of positive training data for Course2vec, since only the good

Table 8.2: Prediction performance of the proposed representation learning based approaches for grade-aware course recommendation.

Metric	SVD		Course2vec	
	(+)	(+-)	(+)	(+-)
Recall(good)	<u>0.468</u>	0.396	0.448	0.351
Recall(bad)	0.372	0.206	0.404	<u>0.202</u>
Recall(diff)	0.096	<u>0.190</u>	0.044	0.149

The underlined entries denote the results with the best performance for each metric.

courses are used as positive examples for learning the models. This is supported by the comparable prediction performance of the (++)-based approaches that use all the available training data as positive examples, which are shown in Table 8.5.

Comparing the (+)- and (+-)-based methods, we see that, the (+-)-based model achieves a worse Recall(good) value, but a much better Recall(bad) value. For instance, SVD(+-) achieves a 15% decrease in Recall(good) and a 45% decrease in Recall(bad) over SVD(+). This is expected, since utilizing the bad course information gives the models more power to learn to rank these courses low, but it also adds some noise, since different students with the same or similar previous set of courses can achieve different outcomes on the same courses.

8.4.2 Comparison of the Grade-aware Recommendation Approaches Combining Grade Prediction with Course Recommendation

Table 8.3 shows the prediction performance of the grade-aware recommendation approaches that combine CKRM with the grade-aware and grade-unaware representation learning methods. The results show that CKRM+SVD(++) achieves the best Recall(good), while CKRM+Course2vec(+-) achieves the best Recall(bad). Overall, CKRM+SVD(+-) achieves the best Recall(diff). Combining CKRM with the grade-unaware, i.e., (++)-based, approaches helped in differentiating between good and bad courses, by achieving a high Recall(diff) of 0.158 and 0.142 for SVD and Course2vec, respectively. However, despite these performance improvements, the combinations that use the grade-aware recommendation methods do better. For instance, CKRM+SVD(+) outperforms CKRM+SVD(++) by 15% in terms of Recall(diff).

The results also show that the SVD-based (+)- and (+-)-based approaches outperform their Course2vec counterparts in terms of Recall(diff), similar to the results of

Table 8.3: Prediction performance of combining CKRM with the representation learning based approaches for grade-aware course recommendation methods.

Metric	CKRM + SVD			CKRM + Course2vec		
	(++)	(+)	(+-)	(++)	(+)	(+-)
Recall(good)	<u>0.438</u>	0.417	0.385	0.411	0.417	0.338
Recall(bad)	0.279	0.230	0.189	0.269	0.264	<u>0.183</u>
Recall(diff)	0.158	0.187	<u>0.197</u>	0.142	0.152	0.155

The underlined entries denote the results with the best performance for each metric.

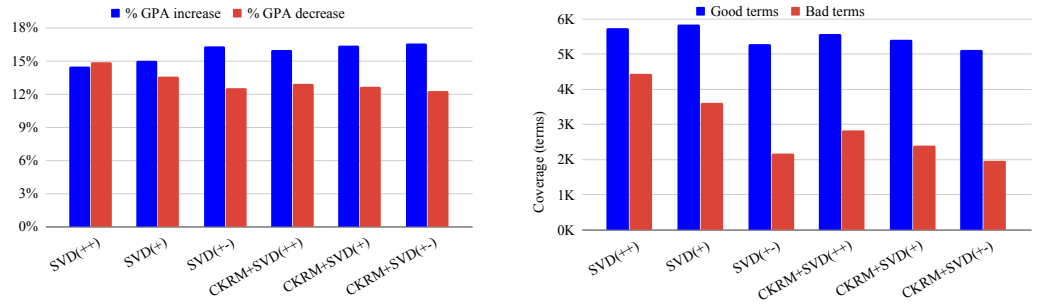
SVD and Course2vec alone (Sec. 8.4.1). Unlike the difference in the performance of SVD(+) vs SVD(+-), CKRM+SVD(+) achieves a similar Recall(diff) to that achieved by CKRM+SVD(+-) (and the same holds for the Course2vec-based approaches). The difference is that CKRM+SVD(+) achieves higher Recall(good) and Recall(bad) than CKRM+SVD(+-).

8.4.3 Comparison of the Proposed Approaches for Grade-aware Course Recommendation

Comparing each of the SVD- and Course2vec-based approaches with and without CKRM (shown in Tables 8.2 and 8.3), we see that combining CKRM with the (+)-based approaches significantly improved their performance with 95% and 245% increase in Recall(diff) for SVD and Course2vec, respectively. On the other hand, combining CKRM with the (+-)-based approaches achieves comparable performance to using the corresponding (+-)-based approach alone.

By further analyzing these ranking results, Figure 8.3 shows the percentage GPA increase and decrease as well as the coverage for good and bad terms for each SVD-based method with and without CKRM[4]. CKRM+SVD(+) outperforms SVD(+) in all but one metric, which is coverage for good terms, where it achieves slightly worse performance than SVD(+). On the other hand, CKRM+SVD(+-) has comparable performance to SVD(+-), which is analogous to their recall metrics results.

[4]The results of the Course2vec-based methods are similar, and are thus omitted.



(a) Percentage GPA increase and decrease

(b) Coverage for good and bad terms

Figure 8.3: Performance of the different SVD-based methods with and without CKRM (refer to Sec. 8.3.3 for the metrics definitions).

8.4.4 Representation Learning vs Competing Approaches for Grade-aware Course Recommendation

Table 8.4 shows the prediction performance of the representation learning and competing approaches for grade-aware course recommendation. Grp-pop(+-) achieves the best Recall(diff) among the three competing (baseline) approaches. The results also show that SVD(+) achieves the best Recall(good), while grp-pop(+-) achieves the best Recall(bad). Overall, SVD(+-) achieves the best Recall(diff).

8.4.5 Grade-aware vs Grade-unaware Representation Learning Approaches

Table 8.5 shows the performance prediction of the representation learning approaches for grade-aware, i.e., (+)- and (+-)-based approaches, and grade-unaware, i.e., (++)-based approach, course recommendation. Each of SVD(+) and Course2vec(+) achieves a Recall(good) that is comparable to or better than that achieved by its corresponding (++)-based approach. In addition, both the (+)- and (+-)-based methods achieve much better (lower) Recall(bad). For instance, SVD(+) and SVD(+-) achieve 0.372 and 0.206 Recall(bad), respectively, resulting in 26% and 59% improvement over SVD(++), respectively.

By comparing the (++)-, (+)-, and (+-)-based approaches in terms of Recall(diff),

Table 8.4: Prediction performance of the representation learning based vs competing approaches for grade-aware course recommendation.

Metric	Dependency Graph	Grp-pop (+)	Grp-pop (+-)	SVD (+)	SVD (+-)	Course2vec (+)	Course2vec (+-)
Recall(good)	0.385	0.425	0.367	<u>0.468</u>	0.396	0.448	0.351
Recall(bad)	0.249	0.343	<u>0.188</u>	0.372	0.206	0.404	0.202
Recall(diff)	0.136	0.082	0.179	0.096	<u>0.190</u>	0.044	0.149

The underlined entries denote the results with the best performance for each metric.

Table 8.5: Prediction performance of the representation learning based approaches for grade-aware and grade-unaware course recommendation.

Metric	SVD (++)	Course2vec (++)	SVD (+)	Course2vec (+)	SVD (+-)	Course2vec (+-)
Recall(good)	0.453	0.455	<u>0.468</u>	0.448	0.396	0.351
Recall(bad)	0.502	0.493	0.372	0.404	0.206	<u>0.202</u>
Recall(diff)	-0.048	-0.038	0.096	0.044	<u>0.190</u>	0.149

The underlined entries denote the results with the best performance for each metric.

we can see that the (++)-based approaches achieve negative recall values which indicates that they recommend more bad courses than they recommend good ones. The (+)-based approaches do slightly better, while the (+-)-based approaches achieve the highest Recall(diff). This is expected, since the (++)-based methods treat both types of subsequent courses equally during their learning, and so they recommend both types in an equal manner. This shows that differentiating between good and bad courses in any course recommendation method is very helpful for ranking the good courses higher than the bad ones, which will help the student maintain or improve their overall GPA.

In terms of percentage GPA increase and decrease (shown in Figure 8.3), SVD(+-) outperforms SVD(++) by 2% in percentage GPA increase and 2.5% in percentage GPA decrease. Moreover, SVD(+-) achieves $\sim$ 62% less coverage for the bad terms than SVD(++), while it achieves $\sim$ 10% less coverage for the good terms.

8.5 Analysis of Recommendation Accuracy

Our discussion so far focused on analyzing the performance of the different methods by looking at metrics that are aggregated across the different majors. However, given that

the structure of the degree programs of different majors is sometimes quite different, and that different student groups can exhibit different characteristics, an important question that arise is how the different methods perform across the individual degree programs and different student groups and if there are methods that consistently perform well across majors as well as across student groups. In this section, we analyze the recommendations done by one of our best performing models, CKRM+SVD(+-), against the best performing baseline, i.e., grp-pop(+-), in terms of Recall(diff), across these degree programs and student groups (RQ6).

Analysis on Different Majors

Table 8.4 shows the recommendation accuracy, in terms of Recall(diff), across the 23 majors, by both grp-pop(+-) and CKRM+SVD(+-) (Fig 8.4a). First, we can see that there is a huge variation in the recall values across the majors, ranging from 0.05 to ∼0.5. Second, we see that CKRM+SVD(+-) consistently outperforms grp-pop(+-), except for the nursing major. To further look into why this happens, we investigated some of the characteristics of the students' degree sequences. For each major, we computed the pairwise percentage of common courses among students who belong to that major, which is shown in Figure 8.4b. In addition, we computed the similarity in the sequencing, i.e., ordering, of the common courses between each pair of students, which is shown in Figure 8.4c. For computing the pairwise degree similarity, we utilized the formula that we proposed in Chapter 7 (see Eq. 7.5).

We found that there is a high correlation between the Recall(diff) values and both the average pairwise percentage of common courses and the average pairwise degree similarity among students of these majors (correlation values of 0.47 and 0.5 for grp-pop(+-), and 0.47 and 0.38 for CKRM+SVD(+-), respectively). This implies that, as the percentage of common courses and degree similarity between pairs of students decrease, accurate course recommendation becomes more difficult, since there is more variability in the set of courses taken as well as their sequencing. The nursing major, where grp-pop(+-) significantly outperforms CKRM+SVD(+-) has the highest average pairwise percentage of common courses, ∼76%, as well as the highest average pairwise degree similarity, ∼0.86, compared to all other majors. This implies that the nursing major is the most restricted major and that students tend to follow highly similar degree

plans and take very similar courses at each academic level. The group popularity ranking in this case can easily outperform other recommendation methods.

Analysis on Different Student Groups

Figure 8.5 shows the recommendation accuracy, in terms of Recall(diff), for grp-pop(+-) and CKRM+SVD(+-) across different student sub-groups.

Figure 8.5a shows the recommendation accuracy among different GPA-based student types, A vs B vs C. We notice that, first, CKRM+SVD(+-) outperforms grp-pop(+-) for all student groups. Second, we found that CKRM+SVD(+-) achieves the highest Recall(diff) for the type-B students, followed by type-A, and then by type-C. This could be due to the following reasons. After analyzing the training data, we found that the type-A and type-B students constitute ∼96% of the student population. After analyzing the average pairwise percentage of common courses and degree similarity among each GPA-based groups of students, as well as among pairs of different GPA-based groups, we found that type-C students follow more diverse sequencing for their degree plans that type-A or type-B students, as illustrated in Table 8.6, while there was no difference among the different groups in the average pairwise percentage of common courses. As discussed in Sec. 8.5, there is a high correlation between the pairwise degree similarity and the recommendation accuracy. Since there is no enough training data for the type-C students to learn their sequencing of the courses, this can explain why the recommendation accuracy for them was the lowest.

Table 8.6: Average pairwise degree similarity between different pairs of GPA-based student groups.

Student Pair	Degree Similarity
A-B	0.597
A-C	0.535
B-C	0.534

The column "Student Pair" denotes the GPA type of the pair of students whose degree similarity was computed.

Figure 8.5b shows the recommendation accuracy among different student sub-groups based on their academic level. At the University of Minnesota, there are four academic levels, based on the number of both earned and transferred credits by the beginning of the semester: (1) freshman ($\leq$ 30 credits), (2) sophomore ($>$ 30 and $\leq$ 60 credits), (3)

junior (> 60 and ≤ 90 credits), and senior (> 90 credits). First, we can notice that CKRM+SVD(+-) significantly outperforms grp-pop(+-) across all student groups. Second we see that, as the student's academic level increases, and hence he/she has spent more years at the university and took more courses, both methods tend to achieve more accurate recommendations. This can be due to the following reasons. First, since we filter out the courses that have been previously taken by the student before making recommendations (see Sec. 8.3.3), this means that as the student's academic level increases, there is a smaller number of candidate courses from which the recommendations are to be made. Second, for CKRM+SVD(+-), as the student takes more courses, his/her implicit profile that is computed by aggregating the embeddings of the previously-taken courses becomes more accurate.

8.6 Characteristics of Recommended Courses

An important question to any recommendation method is what the characteristics of the recommendations are. In this section, we study two important characteristics for the recommended courses; (i) the difficulty of courses (Sec. 8.6.1), and (ii) the popularity of them (Sec. 8.6.2) (RQ7).

8.6.1 Course Difficulty

As our proposed grade-aware recommendation methods are trained to recommend courses that help students maintain or improve their GPA, these methods can be prone to recommending more easier courses in which students usually achieve high grades. Here, we investigate whether this happens in our recommendations or not. Table 8.7 shows the grade statistics of all courses, as well as the courses recommended by all variations of grade-unaware and grade-aware SVD variations. The mean grade is 3.5 for all courses, while for the recommended courses, it is 3.24, 3.4, and 3.56, for SVD(++), SVD(+) and SVD(+-), respectively. These statistics show that the grade-aware SVD approaches tend to only slightly favor easier courses in their recommendations than the grade-unaware SVD approach.

Table 8.7: Statistics for the grades of all and recommended courses.

Course Set	Mean	Median	Std. Dev.
All	3.50	3.61	0.51
SVD(++)	3.24	3.24	0.27
SVD(+)	3.40	3.40	0.24
SVD(+-)	3.56	3.55	0.20

8.6.2 Course Popularity

Since the university administrators need to make sure that students are enrolled in courses with different popularity, as there is a capacity for each course and classroom, course popularity is an important factor for course recommendations.

We also analyze the results of our models in terms of the popularity of the courses they recommend. Figure 8.6.2 shows the frequency of the actual good courses in the test set, as well as the frequency of the good courses recommended by both grp-pop(+-) and CKRM+SVD(+-)[5].

The figure shows that both grp-pop(+-) and CKRM+SVD(+-) recommend courses with different popularity[6], similar to the actual good courses taken by students. Comparing CKRM+SVD(+-) to grp-pop(+-), we can notice that, grp-pop(+-) tends to recommend a higher number of the more popular courses, while CKRM+SVD(+-) recommends more of the less popular ones, which can be considered a major benefit for the latter method.

8.7 Summary

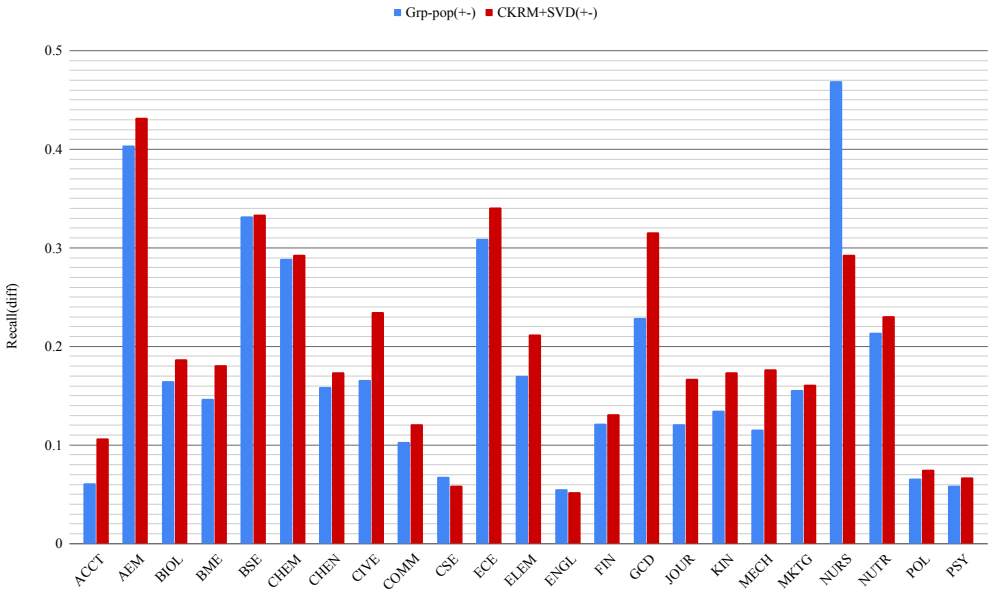
In this chapter, we proposed grade-aware course recommendation approaches for solving the course recommendation problem. The proposed approach aims to recommend to students good courses on which the student's expected grades will maintain or improve their overall GPA. We proposed two different approaches for solving the grade-aware course recommendation problem. The first approach ranks the courses by using an

[5]Remember that we recommend $n_{(s,t)}$ courses, which is the total number of (good and bad) courses taken by student s in term t (see Sec. 8.3.3), so the number of recommendations can be higher than the number of actual good courses.

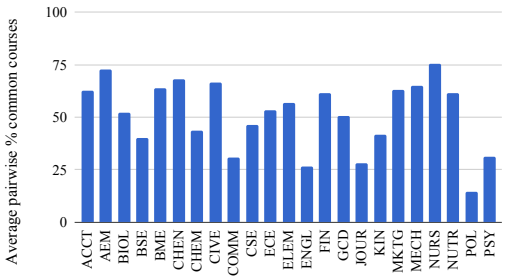
[6]Since we use a filtering technique before making recommendations, grp-pop(+-) can recommend courses with little popularity (see Sec. 8.3.3)

objective function that differentiates between sequences of courses that are expected to increase or decrease a student's GPA. The second approach combines the grades predicted by grade prediction methods in order to improve the rankings produced by course recommendation methods. To obtain course rankings in the first approach, we adapted two widely-known representation learning techniques; one that uses the linear Singular Value Decomposition model, while the other uses log-linear neural network based models.

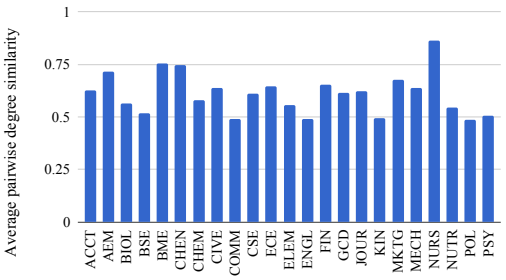
We conducted an extensive set of experiments on a large dataset obtained from 23 different majors at the University of Minnesota. The results showed that: (i) the proposed grade-aware course recommendation approaches outperform grade-unaware recommendation methods in recommending more courses that increase the students' GPA and fewer courses that decrease it; (ii) the proposed representation learning based approaches outperform competing approaches for grade-aware course recommendation; and (iii) the approaches that utilize both the good and bad courses and differentiates between them achieve comparable performance to combining grade prediction with the approaches that either utilize the good courses only, or those that differentiate between good and bad courses.



(a) Per-major recommendation accuracy of grp-pop(+-) and SVD(+-).

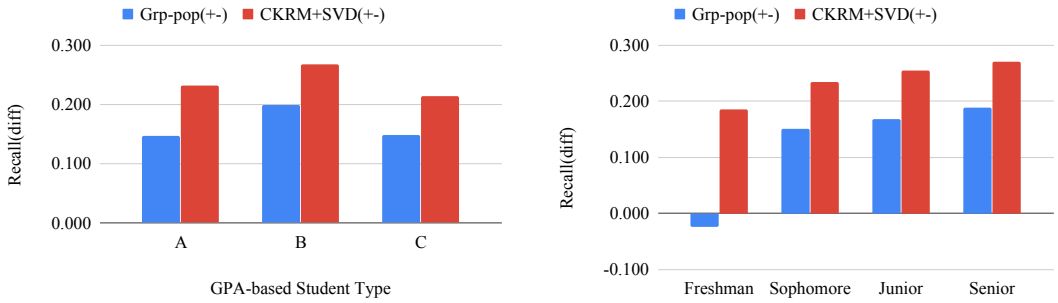


(b) Pairwise % common courses per major.



(c) Pairwise degree similarity per major.

Figure 8.4: Per-major recommendation accuracy and the characteristics of the students' degrees.



(a) Recommendation accuracy per student type. (b) Recommendation accuracy per cohort.

Figure 8.5: Recommendation accuracy of SVD(+-) on different student sub-groups.

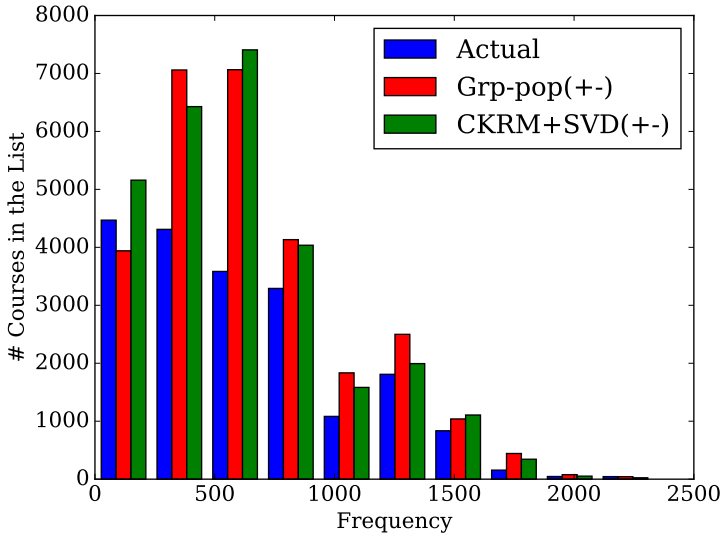


Figure 8.6: Popularity of the actual good courses, as well as courses recommended by grp-pop(+-) and CKRM+SVD(+-).

Chapter 9

Conclusions and Future Directions

9.1 Summary of Contributions

The alarming statistics of student retention and graduation rates have drawn the researchers' interest to apply machine learning and data mining techniques to improve these rates and help current and future students in selecting appropriate courses for them that match their skills and backgrounds and help them towards graduating in a successful and timely manner. Towards this end, several grade prediction and course recommendation approaches have been recently developed that utilize different machine learning techniques as well as domain-specific approaches.

This thesis focused on the development of novel data-driven methods to improve the accuracy of grade prediction and course recommendation and derive useful insights from the actual students' data that can help in better designing degree plans and pre-requisite charts.

Our main contributions in this thesis are as follows:

- **Developing cumulative knowledge-based regression models for grade prediction.** Previous grade prediction methods build on the idea that students accumulate knowledge over time and that their grades in previous courses affect their grades in future ones. One limitation of such previous methods is that they

treat each course independently from each other, assuming that each course provides a unique set of knowledge components. In more flexible degree programs, that offer a variety of courses that provide overlapping knowledge, such methods can suffer from generalization. Another limitation is that they require sufficient training data for each (prior, target) course pair in order to learn accurate regression models. We developed a more generalized set of cumulative knowledge-based regression models (CKRM) that project all courses into a unified knowledge component space, which can be either a latent or textual-based space. The experimental evaluation of these methods, performed on a large real-world dataset, showed that CKRM outperforms previous grade prediction methods, especially in the more flexible degree programs. In addition, the textual-based CKRM methods revealed some interesting insights about the hidden pre-requisite keywords for courses that belong to unlisted pre-requisite courses to them.

- **Developing context-aware non-linear and neural attentive grade prediction methods**. The previously-proposed CKRM method learns shallow linear models that may not be able to capture the complex interactions among prior courses. Moreover, previous grade prediction methods ignored the effect of concurrently-taken courses when predicting a student's grade in a specific course. We developed context-aware non-linear and neural attentive models that: (i) model the complex interactions among prior courses, by utilizing maximum knowledge-based and neural attentive models; and (ii) model the interaction between a target course and concurrently-taken courses by estimating a context-aware embedding for the target course. To learn the attention weights in the neural attentive models, we utilized the commonly-used softmax activation function, as well as the newly-proposed sparsemax activation function, that can assign zero attention to the irrelevant courses. A comprehensive set of experiments on a large real-world dataset showed that the proposed context-aware non-linear and neural attentive models improved the prediction accuracy, with statistical significant improvements over the competing grade prediction methods. In addition, a qualitative analysis on the sparse attention weights learned by the neural attentive models showed that they were able to uncover hidden prerequisites for target courses, which can be useful for degree planning and course sequencing.

- **Studying the relationship between course timing and ordering of the students' degree plans and their GPAs and time to degree.** Student success in undergraduate education is measured by both his/her GPA and time to degree. Different variables, such as family background, prior academic achievement, and working status, have been explored in previous studies on how they affect the student's success. However, degree planning has not been studied before. To gain deeper insights about how the time when students take their courses and how they order their courses can be related to their GPAs and time to degree, we conducted a large-scale analysis by defining metrics to measure course timing and sequencing and comparing their values among different GPA- and time-to-degree-based groups of students. The analysis, done on a large real-world dataset, showed that course timing and ordering is more correlated to the students' time to degree than to their GPAs. In addition, we performed a case study on time to degree prediction using new course timing and ordering features that were shown to outperform other baseline features used in previous studies.

- **Developing grade-aware course recommendation approaches.** To help students in their course selection each term, both course recommendation and grade prediction methods can be used. Course recommendation focuses on learning course sequence patterns to recommend to students courses that can help them towards finishing their degrees. Grade prediction focuses on accurately predicting the students' grades in courses they would like to take. Each problem has been studied separately in most previous studies. We developed a grade-aware course recommendation framework that aims to recommend to students courses that help them both finish their degrees in a timely manner and maintain or improve their overall GPAs, by combining the benefits of both course recommendation and grade prediction. We developed two main approaches for grade-aware course recommendation: one that explicitly differentiates between the courses that help maintain or improve the students' GPAs and the courses that decrease them, and the other that combines the results of both course recommendation and grade prediction models in a non-linear way. A comprehensive set of experiments, performed on a large real-world dataset, showed that the proposed grade-aware course recommendation approaches can better help students by recommending courses that are

expected to maintain or improve their GPAs. In addition, the proposed approaches outperformed other competing grade-aware course recommendation approaches.

9.2 Future Research Directions

In this thesis, we have developed data-driven methods for grade prediction and course recommendation to better assist students during the process of course selection and help improve student retention and graduation rates. Here we outline some future research directions that stem from our work.

Along with the students' grades data, there is additional data available that can help in designing better grade prediction and course recommendation approaches. For instance, there is students' demographic data, instructors for courses, degree requirements, as well as the students' interactions in the Learning Management System of the university, such as Moodle. In the recent years, some methods have been proposed that incorporate some of these types of features separately. However, there is still a lot of scope to utilize all this rich data to better learn course recommendation and grade prediction models.

As the students' success in the university is measured by both their GPAs and time to degree, it is important to consider both these factors when recommending courses to students. Some previous studies have tackled the problem of course sequence recommendation, but these methods either did not take the student's GPA into consideration, or depend on an exact extraction system for the degree plans that makes it hard to suggest course sequences when there is not enough training data.

The methods developed in this thesis are all offline methods, meaning that they cannot learn from their mistakes. An interesting research direction would be to let the students use the predicted grades and check the courses recommended to them, and after each semester, the errors or mistakes done are logged into the system. Analyzing and incorporating these errors while developing grade prediction and course recommendation methods should help improve their accuracy much better.

References

[1] Grace Kena, William Hussar, Joel McFarland, Cristobal de Brey, Lauren Musu-Gillette, Xiaolei Wang, Jijun Zhang, Amy Rathbun, Sidney Wilkinson-Flicker, Melissa Diliberti, et al. The condition of education 2016. nces 2016-144. *National Center for Education Statistics*, 2016.

[2] John M Braxton, Amy S Hirschy, and Shederick A McClendon. *Understanding and Reducing College Student Departure: ASHE-ERIC Higher Education Report, Volume 30, Number 3*, volume 16. John Wiley & Sons, 2011.

[3] Alejandro Peña-Ayala, editor. *Learning Analytics: Fundaments, Applications, and Trends*, volume 94. Springer International Publishing, 2017.

[4] Agoritsa Polyzou and George Karypis. Grade prediction with course and student specific models. In *PAKDD*. Springer, 2016.

[5] Mack Sweeney, Jaime Lester, Huzefa Rangwala, and Aditya Johri. Next-term student performance prediction: A recommender systems approach. *Journal of Educational Data Mining*, 8(1):22–51, 2016.

[6] Annika Wolff, Zdenek Zdrahal, Andriy Nikolov, and Michal Pantucek. Improving retention: predicting at-risk students by analysing clicking behaviour in a virtual learning environment. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 145–149. ACM, 2013.

[7] Annika Wolff, Zdenek Zdrahal, Drahomira Herrmannova, Jakub Kuzilek, and Martin Hlosta. Developing predictive models for early detection of at-risk students on distance learning modules. 2014.

[8] Elvira Lotsari, Vassilios S Verykios, Chris Panagiotakopoulos, and Dimitris Kalles. A learning analytics methodology for student profiling. In *Hellenic Conference on Artificial Intelligence*, pages 300–312. Springer, 2014.

[9] Dragan Gasevic, Vitomir Kovanovic, Srecko Joksimovic, and George Siemens. Where is research on massive open online courses headed? a data analysis of the mooc research initiative. *The International Review of Research in Open and Distributed Learning*, 15(5), 2014.

[10] Asmaa Elbadrawy, R Scott Studham, and George Karypis. Collaborative multi-regression models for predicting students' performance in course activities. In *LAK*, 2015.

[11] Siddharth Reddy, Igor Labutov, and Thorsten Joachims. Latent skill embedding for personalized lesson sequence recommendation. *arXiv preprint*, 2016.

[12] Andrew S Lan, Andrew E Waters, Christoph Studer, and Richard G Baraniuk. Sparse factor analysis for learning and content analytics. *The Journal of Machine Learning Research*, 2014.

[13] José P González-Brenes and Jack Mostow. Dynamic cognitive tracing: Towards unified discovery of student and cognitive models. In *EDM*, 2012.

[14] Everaldo Aguiar, Nitesh V Chawla, Jay Brockman, G Alex Ambrose, and Victoria Goodrich. Engagement vs performance: using electronic portfolios to predict first semester engineering student retention. In *Proceedings of the Fourth International Conference on Learning Analytics And Knowledge*, pages 103–112. ACM, 2014.

[15] Raheela Asif, Agathe Merceron, and Mahmood Khan Pathan. Investigating performance of students: a longitudinal study. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 108–112. ACM, 2015.

[16] Asmaa Elbadrawy and George Karypis. Domain-aware grade prediction and top-n course recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 183–190. ACM, 2016.

[17] Narimel Bendakir and Esma Aïmeur. Using association rules for course recommendation. In *Proceedings of the AAAI Workshop on Educational Data Mining*, volume 3, 2006.

[18] Youngseok Lee and Jungwon Cho. An intelligent course recommendation system. *SmartCR*, 1(1):69–84, 2011.

[19] Aditya G Parameswaran and Hector Garcia-Molina. Recommendations with prerequisites. In *Proceedings of the third ACM conference on Recommender systems*, pages 353–356. ACM, 2009.

[20] Aditya G Parameswaran, Georgia Koutrika, Benjamin Bercovitz, and Hector Garcia-Molina. Recsplorer: recommendation algorithms based on precedence mining. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 87–98. ACM, 2010.

[21] Aditya G Parameswaran, Hector Garcia-Molina, and Jeffrey D Ullman. Evaluating, combining and generalizing recommendations with prerequisites. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 919–928. ACM, 2010.

[22] Aditya Parameswaran, Petros Venetis, and Hector Garcia-Molina. Recommendation systems with complex constraints: A course recommendation perspective. *ACM TOIS*, 29(4):20, 2011.

[23] Zachary A Pardos, Zihao Fan, and Weijie Jiang. Connectionist recommendation in the wild: on the utility and scrutability of neural networks for personalized course guidance. *User Modeling and User-Adapted Interaction*, pages 1–39, 2019.

[24] Michael Backenköhler, Felix Scherzinger, Adish Singla, and Verena Wolf. Data-driven approach towards a personalized curriculum. In *Proceedings of the 11th International Conference on Educational Data Mining*, pages 246–251, 2018.

[25] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. *Numerische mathematik*, 14(5):403–420, 1970.

[26] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Application of dimensionality reduction in recommender system a case study. In *Proceeding of WebKDD-2000 Workshop*, 2000.

[27] Robert Bell, Yehuda Koren, and Chris Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 95–104, New York, NY, USA, 2007. ACM.

[28] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8, 2007.

[29] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[30] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[31] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[32] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics, 2012.

[33] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[34] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196, 2014.

[35] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[36] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077. International World Wide Web Conferences Steering Committee, 2015.

[37] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

[38] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. Playlist prediction via metric embedding. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 714–722. ACM, 2012.

[39] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. E-commerce in your inbox: Product recommendations at scale. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1809–1818. ACM, 2015.

[40] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 403–412. ACM, 2015.

[41] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, 2016.

[42] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[43] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. Attentional factorization machines: learning the weight of feature interactions via attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 3119–3125. AAAI Press, 2017.

[44] Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 355–364. ACM, 2017.

[45] Lei Mei, Pengjie Ren, Zhumin Chen, Liqiang Nie, Jun Ma, and Jian-Yun Nie. An attentive interaction network for context-aware recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 157–166. ACM, 2018.

[46] Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. Nais: Neural attentive item similarity model for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 30(12):2354–2366, 2018.

[47] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 335–344. ACM, 2017.

[48] Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning*, pages 1614–1623, 2016.

[49] Anirban Laha, Saneem Ahmed Chemmengath, Priyanka Agrawal, Mitesh Khapra, Karthik Sankaranarayanan, and Harish G Ramaswamy. On controllable sparse alternatives to softmax. In *Advances in Neural Information Processing Systems*, pages 6423–6433, 2018.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[51] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[52] Zhiyun Ren, Xia Ning, and Huzefa Rangwala. Ale: Additive latent effect models for grade prediction. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 477–485. SIAM, 2018.

[53] Nina Hagemann, Michael P OMahony, and Barry Smyth. Module advisor: Guiding students with recommendations. In *International Conference on Intelligent Tutoring Systems*, pages 319–325. Springer, 2018.

[54] Kiratijuta Bhumichitr, Songsak Channarukul, Nattachai Saejiem, Rachsuda Jiamthapthaksin, and Kwankamol Nongpong. Recommender systems for university elective course recommendation. In *Computer Science and Software Engineering (JCSSE), 2017 14th International Joint Conference on*, pages 1–5. IEEE, 2017.

[55] Mihai Cucuringu, Charles Z Marshak, Dillon Montag, and Puck Rombach. Rank aggregation for course sequence discovery. In *International Workshop on Complex Networks and their Applications*, pages 139–150. Springer, 2017.

[56] Serge Herzog. Estimating student retention and degree-completion time: Decision trees and neural networks vis-à-vis regression. *New directions for institutional research*, 2006(131):17–33, 2006.

[57] Carmen Aina, Eliana Baici, and Giorgia Casalone. Time to degree: students' abilities, university characteristics or something else? evidence from italy. *Education Economics*, 19(3):311–325, 2011.

[58] Agar Brugiavini, Carlo Carraro, and Matija Kovacic. Academic achievements: Grades versus duration. *International Center for Climate Governance*, 2015.

[59] Katja Theune. The working status of students and time to degree at german universities. *Higher Education*, 70(4):725–752, 2015.

[60] Andreas Behr and Katja Theune. The causal effect of off-campus work on time to degree. *Education Economics*, 24(2):189–209, 2016.

[61] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[62] Sara Morsy and George Karypis. Cumulative knowledge-based regression models for next-term grade prediction. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 552–560. SIAM, 2017.

[63] Qian Hu and Huzefa Rangwala. Course-specific markovian models for grade prediction. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 29–41. Springer, 2018.