A Machine Learning Approach for Tracking and Predicting Student Performance in Degree Programs

Jie Xu, *Member, IEEE,* Kyeong Ho Moon, *Student Member, IEEE,* and Mihaela van der Schaar, *Fellow, IEEE*

***Abstract*—Accurately predicting students' future performance based on their ongoing academic records is crucial for effectively carrying out necessary pedagogical interventions to ensure students' on-time and satisfactory graduation. Although there is a rich literature on predicting student performance when solving problems or studying for courses using data-driven approaches, predicting student performance in completing degrees (e.g. college programs) is much less studied and faces new challenges: (1) Students differ tremendously in terms of backgrounds and selected courses; (2) Courses are not equally informative for making accurate predictions; (3) Students' evolving progress needs to be incorporated into the prediction. In this paper, we develop a novel machine learning method for predicting student performance in degree programs that is able to address these key challenges. The proposed method has two major features. First, a bilayered structure comprising of multiple base predictors and a cascade of ensemble predictors is developed for making predictions based on students' evolving performance states. Second, a data-driven approach based on latent factor models and probabilistic matrix factorization is proposed to discover course relevance, which is important for constructing efficient base predictors. Through extensive simulations on an undergraduate student dataset collected over three years at UCLA, we show that the proposed method achieves superior performance to benchmark approaches.**

***Index Terms*—Student performance prediction, data-driven course clustering, personalized education**

I. Introduction

Making higher education affordable has a significant impact on ensuring the nation's economic prosperity and represents a central focus of the government when making education policies [1]. Yet student loan debt in the United States has blown past the trillion-dollar mark, exceeding Americans' combined credit card and auto loan debts [2]. As the cost in college education (tuitions, fees and living expenses) has skyrocketed over the past few decades, prolonged graduation time has become a crucial contributing factor to the ever-growing student loan debt. In fact, recent studies show that only 50 of the more than 580 public four-year institutions in the United States have on-time graduation rates at or above 50 percent for their full-time students [2].

To make college more affordable, it is thus crucial to ensure that many more students graduate on time through early interventions on students whose performance will be unlikely to meet the graduation criteria of the degree program on time. A critical step towards effective intervention is to build a system that can continuously keep track of students' academic performance and accurately predict their future performance, such as when they are likely to graduate and their estimated final GPAs, given the current progress. Although predicting student performance has been extensively studied in the literature, it was primarily studied in the contexts of solving problems in Intelligent Tutoring Systems (ITSs) [3][4][5][6], or completing courses in classroom settings or in Massive Open Online Courses (MOOC) platforms [7][8]. However, predicting student performance within a degree program (e.g. college program) is significantly different and faces new challenges.

First, students can differ tremendously in terms of backgrounds as well as their chosen areas (majors, specializations), resulting in different selected courses as well as course sequences. On the other hand, the same course can be taken by students in different areas. Since predicting student performance in a particular course relies on the student past performance in other courses, a key challenge for training an effective predictor is how to handle heterogeneous student data due to different areas and interests. In contrast, solving problems in ITSs often follow routine steps which are the same for all students [9]. Similarly, predictions of students' performance in courses are often based on in-course assessments which are designed to be the same for all students [7].

Second, students may take many courses but not all courses are equally informative for predicting students' future performance. Utilizing the student's past performance in all courses that he/she has completed not only increases complexity but also introduces noise in the prediction, thereby degrading the prediction performance. For instance, while it makes sense to consider a student's grade in the course "Linear Algebra" for predicting his/her grade in the course "Linear Optimization", the student's grade in the course "Chemistry Lab" may have much weaker predictive power. However, the course correlation is not always as obvious as in this case. Therefore, discovering the underlying correlation among courses is of great importance for making accurate performance predictions.

Third, predicting student performance in a degree program is not a one-time task; rather, it requires continuous tracking and updating as the student finishes new courses over time. An important consideration in this regard is that the prediction needs to be made based on not only the most recent snapshot of the student accomplishments but also the evolution of the student progress, which may contain valuable information for

J. Xu is with the Department of Electrical and Computer Engineering, University of Miami, Coral Gables, FL, USA.

K. H. Moon and M. van der Schaar are with the Department of Electrical Engineering, University of California, Los Angeles, USA. M. van der Schaar is also with Oxford-Man Institute of Quantitative Finance and the Department of Engineering Science at Oxford University, UK. Their research is supported by NSF under grants ECCS 1407712 and PFI:BIC 1533983.

making more accurate predictions. However, the complexity can easily explode since even mathematically representing the evolution of student progress itself can be a daunting task. However, treating the past progress equally as the current performance when predicting the future may not be a wise choice either since intuition tells us that old information tends to be outdated.

In light of the aforementioned challenges, in this paper, we propose a novel method for predicting student performance in a degree program. We focus on predicting students' GPAs but the general framework can be used for other student performance prediction tasks. Our main contributions are three-fold.

(1) We develop a novel algorithm for making predictions based on students' progressive performance states. It adopts a bilayered structure comprising a base predictor layer and an ensemble predictor layer. In the base layer, multiple base predictors make local predictions given the snapshot of the student's current performance state in each academic term. In the ensemble layer, an ensemble predictor issues a prediction of the student's future performance by synthesizing the local predictions results as well as the previous-term ensemble prediction. The cascading of ensemble predictor over academic terms enables the incorporation of students' evolving progress into the prediction while keeping the complexity low. We also derive a performance guarantee for our proposed algorithm.

(2) We develop a data-driven course clustering method based on probabilistic matrix factorization, which automatically outputs course clusters based on large, heterogeneous and sparse student course grade data. Base predictors are trained using a variety of state-of-the-art machine learning techniques based on the discovered course clustering results. Specifically, only relevant courses in the same cluster are used as input to the base predictors. This not only reduces the training complexity but also removes irrelevant information and reduces noise in making the prediction.

(3) We perform extensive simulation studies on an undergraduate student dataset collected over three years across 1169 students at the Mechanical and Aerospace Engineering department at UCLA. The results show that our proposed method is able to significantly outperform benchmark methods while preserving educational interpretability.

The rest of this paper is organized as follows. Section II discusses the related work. Section III formulates the student performance prediction problem and provides an overview of the proposed method. Section IV describes how to discover course correlations and train the term-wise base predictors. Section V proposes a novel progressive prediction algorithm to incorporate students' evolving performance into the prediction. Section VI presents the dataset and experimental results. Section VII concludes the paper.

II. Related Work

A. Student Performance Prediction

Machine learning for education has gained much attention in recent years. A substantial amount of literature focuses on predicting student performance in solving problems or completing courses [10]. Many machine learning techniques, such as decision trees [11], artificial neural networks [12], matrix factorization [13], collaborative filters [14] and probabilistic graphical models [15][6], have been applied to develop prediction algorithms. Most of this work ignores the temporal/sequential effect that students improve their knowledge over time and treats the prediction as a one-time task. To take the temporal/sequential effect into account, a three-mode tensor factorization (on student/problem/time) technique was developed for predicting student performance in solving problems in ITSs [16] and a similarity-based algorithm was proposed to issue predictions of student grades in courses only when a certain confidence level is reached [17]. However, due to the aforementioned substantial differences of predicting student performance in degree programs, these methods are not applicable in our setting.

Our progressive prediction algorithm uses the ensemble learning technique, in particular, the Exponentially Weighted Average Forecaster (EWAF) [18] as a building block to enable progressive prediction of student performance and online updating of the predictor as new student data is received. The major difference from the conventional EWAF algorithm is that an ensemble predictor has access to multiple base predictors (experts) as well as the previous-term ensemble predictor, whose output summarizes the outputs of all previous-term base predictors (experts) whereas the conventional EWAF algorithm has access to all experts directly. To our best knowledge, this is a novel architecture for designing predictors for progressively expanding input spaces, which significantly reduces design and implementation complexity and easily scales with the number of academic terms. In this setting, we prove that each ensemble predictor still performs asymptotically no worse than the best base predictor in hindsight among all previous-term base predictors in the worst case, thereby providing strong performance guarantee. More importantly, when the best base predictor is biased towards current-term base predictors, our algorithm is able to achieve better expected regret than the conventional method that has access to all experts directly and treats them equally.

B. Course Relevance Discovery

Our course relevance discovery method is based on the latent factor model [19] and uses the probabilistic matrix factorization algorithm [20] to perform course clustering, which are extensively used in recommender systems [21][22][23]. The problem faced by recommender systems is similar to that for student performance prediction: the dataset in recommender systems is sparse in the sense that each user has rated only a small set of items in the entire item space whereas in our case, each student has taken only a small set of courses in the entire course space. The latent factor model is therefore used to discover the hidden latent factor that resolves the sparsity problem. While recommender systems use the discovered latent factor to enable similarity matching among users and make item recommendations, our system uses the discovered latent factor to cluster relevant courses. It is worth noting that, in the learning context, sparse factor analysis is used to estimate a learner's knowledge of the concepts underlying a domain and the relationships among a collection of questions and those concepts [24].

In [25], the authors make a different connection between recommender systems and student performance prediction. They develop a collaborative filtering algorithm, which is used in recommender systems to recommend items to users based on user similarity. In this paper, we do not develop collaborative filtering prediction algorithms, although they can be adopted as base predictors in our method. More broadly, there is a rich literature on recommending relevant courses or problems to students based on their associated knowledge level, learning styles, and feedbacks [26][27][28]. Course sequence recommendation, which considers the specific course constraints, was studied in [29]. To utilize logged data for course sequence recommendations and curriculum design, an off-policy estimator was developed to estimate how an unobserved policy performs given an observed policy [30]. A rank aggregation framework is adapted for the discovery of optimal course sequences at the university level [31]. However, whereas this literature aims to recommend courses/course sequences based on student backgrounds and past performance, the purpose of the current work is to predict future performance based on student backgrounds and past performance for a given curriculum.

III. Problem Formulation

A. System Model

We consider a degree program in which students must complete a set of courses to graduate in T academic terms. Courses have prerequisite dependencies, namely a course can be taken only when certain prerequisite courses have been taken and passed. In general, the prerequisite dependency can be described as a directed acyclic graph (DAG) (see Figure 1 for an example). There can be multiple specialized areas in a program which require different subsets of courses to be completed for students to graduate. We will focus on the prediction problem for one area in this department. Nevertheless, data from other areas will still be utilized for our prediction tasks. The reason is that data from a single area is often limited while different areas still share many common courses.

Figure 1 illustrates part of the prerequisite graph of the undergraduate program in the Mechanical and Aerospace Engineering (MAE) department at UCLA. In this department, there are two areas, namely Mechanical Engineering (ME) and Aerospace Engineering (AE). Since the complete prerequisite graph is huge, Figure 1 only lists some lower-level courses and a few higher-level courses. As we can see, ME and AE areas share most of the lower level courses but they have distinct upper-level courses.

Students start the program with different backgrounds (e.g. high school GPAs and SAT scores). Denote student i's backgrounds by $\theta_i \in \Theta$ where Θ is the space that includes all possible backgrounds. Student backgrounds are considered as the *static* features of students which do not change as the students advance in the program. In each term t, student i takes a subsets of courses $\mathcal{S}_i^t \subset \mathcal{J}$ and receives a grade

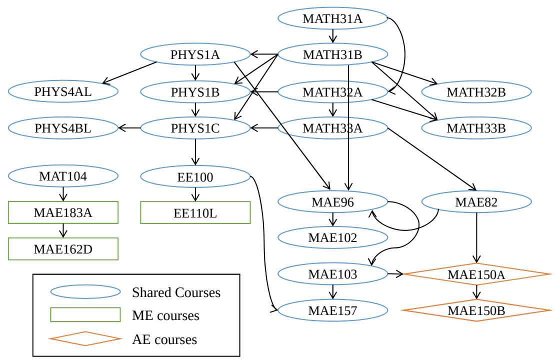


Fig. 1. Part of the prerequisite graph of MAE program at UCLA.

$x_i(j)$ for each of these courses $j \in \mathcal{S}_i^t$. The courses that student i has taken by term t are denoted by $\bar{\mathcal{S}}_i^t = \cup_{\tau=1}^t \mathcal{S}_i^\tau$. The performance state of student i at the end of term t is represented by $\boldsymbol{x}_i^t \in \mathcal{X}_i^t \triangleq [x_{\text{min}}, x_{\text{max}}]^{|\bar{\mathcal{S}}_i^t|}$ where each entry corresponds to the grade of the course taken so far. Therefore, the performance state is expanding over time, namely $\boldsymbol{x}_i^t \preceq \boldsymbol{x}_i^{t+1}, \forall t = 1, 2, ...$ We assume the 4.0 GPA scale. The performance state is considered as the *dynamic* feature of students which is evolving over time. The department usually recommends for each area a sequence in which the students should take the courses. In reality, students mostly follow the recommended sequence of core courses. Hence, by disregarding the elective courses, $\mathcal{S}_i^t$ is more or less the same for all students having the same area. In this paper, we will focus on the core courses and neglect the elective courses which can be vastly different across students. Predicting performance in elective courses will be our future work.

B. Objective

Our objective is to predict the final cumulative GPA (of the core courses) of a student in a certain area at the end of each term t. Specifically, at the end of each term t, the predictor outputs a prediction $\widehat{\text{GPA}}_i^t = F^t(\theta_i, \boldsymbol{x}_i^1, \boldsymbol{x}_i^2, ..., \boldsymbol{x}_i^t)$ given student i' backgrounds θ_i and the evolving performance $\boldsymbol{x}_i^1, \boldsymbol{x}_i^2, ..., \boldsymbol{x}_i^t$. However, because the cumulative GPA is a function of all course grades, namely the weighted average of all course grades $\text{GPA}_i = \sum_{j\in\mathcal{J}} c(j)x_i(j) / \sum_{j\in\mathcal{J}} c(j)$, where $c(j)$ is the credit of course j, the GPA prediction will be more accurate by splitting the known course grades and the unknown course grades. Therefore, the final GPA can be predicted by

$$\widehat{\text{GPA}}_i^t = \frac{\sum_{j\in\bar{\mathcal{S}}^t} c(j)x_i(j) + \sum_{j\in\mathcal{J}\setminus\bar{\mathcal{S}}^t} c(j)\hat{x}_i(j)}{\sum_{j\in\mathcal{J}} c(j)} \tag{1}$$

In other words, instead of predicting the final GPA directly, we will focus on predicting the grade of every course $j \in \mathcal{J}^* \setminus \bar{\mathcal{S}}^t$ that has not been taken by the student yet where $\mathcal{J}^*$ is the set of required courses by the area under consideration. Let $\hat{y}_i(j) = f_j^t(\theta_i, \boldsymbol{x}_i^1, \boldsymbol{x}_i^2, ..., \boldsymbol{x}_i^t)$ denote the predicted grade of course j given the student's backgrounds and evolving performance state up to term t. Note that $\hat{y}_i(j) = \hat{x}_i(j)$ in equation (1). We use different letters to better differentiate what is known (i.e. $\boldsymbol{x}_i^t$) and what is to be predicted (i.e. $y_i^t(j)$) in a given term t. We also highlight that the predictor is using not only a single performance state $\boldsymbol{x}_i^t$ but all the performance states in the previous terms $\boldsymbol{x}_i^1, ..., \boldsymbol{x}_i^{t-1}$, thereby incorporating the evolution of the student's performance. However, the input space of the predictor grows exponentially in the number of terms, thereby making the predictor construction even more challenging. For expositional brevity, we neglect the index j for the targeted course in f_j^t whenever it is clear.

C. Overview of the Proposed Method

In this paper, we propose a novel method for designing predictors based on the evolving progress of students. The key idea is that since the input $\boldsymbol{x}_i^t$ of predictor f^t for term t is a superset of the input $\boldsymbol{x}_i^{t-1}$ of predictor f^{t-1} for term $t-1$, f^t can capitalize on the prediction output $\hat{y}_i^{t-1}$ of f^{t-1} by incorporating the incrementally new information $\boldsymbol{x}_i^t$. This significantly reduces the complexity of constructing f^t and makes the prediction algorithm scalable.

Our approach to enable such progressive predictions is based on the ensemble learning technique and integrates offline learning and online learning. The proposed architecture consists of two layers — a base prediction layer and an ensemble prediction layer.

(1) In the base prediction layer, we construct a set of base predictors $\mathcal{H}$ implemented using different prediction algorithms. For each base predictor $h \in \mathcal{H}$, let $z_{h,i}^t = h(\theta_i, \boldsymbol{x}_i^t)$ denote the prediction result of h for student i given the student's static feature and the current performance state $\boldsymbol{x}_i^t$. The base predictors are trained using a dataset consisting of all student data in the department without differentiating areas to maximally utilize the data. In fact, predictor h may even be trained differently for each term t's prediction task. Therefore, we write $h^t(\theta_i, \boldsymbol{x}_i^t)$ rather than $h(\theta_i, \boldsymbol{x}_i^t)$. Learning the base predictors is done **offline**.

(2) In the ensemble prediction layer, we construct an ensemble predictor for each term. The ensemble predictor f^t for term t synthesizes the previous ensemble output $\hat{y}_i^{t-1}$ and output of the base predictors $z_{h,i}^t, \forall h \in \mathcal{H}^t$ and makes a final prediction $\hat{y}_i^t$ based on $\hat{y}_i^{t-1}$ and $z_{h,i}^t, \forall h \in \mathcal{H}^t$. The ensemble predictor is learned using student data in the same area since students having different areas take different courses and in different sequences and hence, the temporal correlation is likely to be different. Learning the ensemble predictors is done **online**.

A system block diagram of the proposed bilayered architecture for the term t ensemble learning is illustrated in Figure 2. Although the proposed architecture is easy to understand, a couple of challenges must be addressed in order to achieve good prediction performance. The first challenge is how to construct the base predictors. Although existing off-the-shelf machine learning algorithms can be used to perform the prediction task, we would like to construct a base predictor that is customized to the considered course grade prediction problem to improve its prediction performance. A specific consideration in this regard is what information should be included in the training of the predictor as well as making the prediction. The second challenge is how to construct the ensemble predictors and take the temporal correlation into account. Specifically, this is to answer how to synthesize the prediction results of the multiple base predictors as well as the prediction from the previous term. In the next sections, we will develop novel methods to address these challenges.

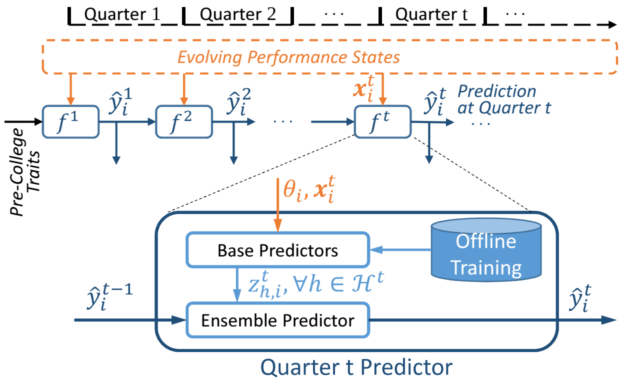


Fig. 2. System Block Diagram.

IV. Offline Learning Base Predictors

In this section, we describe how to learn a base predictor h. In fact, we will learn term-wise base predictors $h^t, \forall t$, which are the same predictor h trained on different training datasets. Specifically, to learn h^t, we will utilize a training dataset $\mathcal{D}^t = \{\boldsymbol{x}_i^t, y_i\}_{i \in \mathcal{I}}$ that contains the performance state $\boldsymbol{x}_i^t$ up to term t for a number of students who have graduated and earned grades y_i for the targeted course. Note that the students, although they are in the same department, are in different areas. One may also wonder why not simply train a single base predictor h based on $\mathcal{D}^T$ and use it for every term. This is because in term $t < T$, much information in $\mathcal{D}^T$ is unavailable yet which may negatively affect the prediction accuracy given the current performance $\boldsymbol{x}_i^t$.

A. Method Overview

An important question when training h^t is how to construct the (input) feature vector given the student performance states $\boldsymbol{x}_i^t, \forall i \in \mathcal{I}$. Because students come from different areas as well as have different interests, the courses in the performance states can be very different. A straightforward way is to construct a large feature vector that contains the grade of courses that have appeared in $\mathcal{D}^t$. Entries corresponding to courses that a student did not take in this vector are filled with null values. In this way, students have the same feature vector format. However, there are two major drawbacks for this method. First, the feature vector can be very large, especially in the later terms of the program when students have taken more courses. The problem is more severe since even though students in different areas may have many courses in common, they also have considerably many distinct courses. In addition to the increased complexity, the second drawback is the possible degraded prediction accuracy due to added noise since not all courses, even the courses within the same area, have predictive power for predicting the grade of the targeted course. Therefore, we will learn the set of courses that are

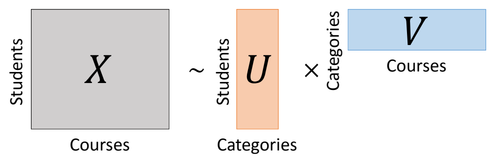


Fig. 3. Illustration of matrix factorization.

more relevant to the targeted course. Notice that for different targeted courses, the relevant courses will also be different. Once the relevant courses are found, the feature vector is constructed using only elements in $\boldsymbol{x}_i^t$ that corresponds to the relevant courses. Then our method will utilize various state-of-the-art supervised learning algorithms to train the base predictors. In this paper, we do not invent new supervised learning algorithms but only focus on learning the relevant courses.

B. Learning Relevant Courses

One way to determine the relevant courses is by using the educational domain knowledge. For instance, the prerequisite dependencies of courses can be used to determine the relevance of different courses. Suppose we want to predict for course j, then only courses that are prerequisites of j are considered as the input courses.

Our method in this paper adopts a data-driven approach to learn the relevant courses in addition to utilizing the educational domain knowledge. We aim to find course clusters where relevant courses are grouped in the same cluster. To this end, we construct a matrix X of size $I \times J$ based on $\mathcal{D}^T$ where the rows represent students and the columns represent courses that appeared in $\mathcal{D}^T$. We aim to find a factorization on X such that $X = U^T V$ where U is the compressed grade matrix of size $K \times I$ and V is the course-cluster matrix of size $K \times J$. The physical meaning of this factorization is as follows. K is the number of course clusters that we try to find. The matrix V represents how the courses are grouped in these K clusters. In particular, $V_{k,j}$ represents the relevance weight of course j with respect to cluster k. A higher value of $V_{k,j}$ means that course j is more likely to be grouped in cluster k. The matrix U represents the students' performance (grades) in each cluster. In particular, $U_{k,i}$ represents student i's average performance (grade) in courses belonging to cluster k. Figure 3 illustrates the idea of matrix factorization.

The above problem can be solved using Singular Value Decomposition (SVD), which aims to minimize $\|X - U^T V\|$. However, student grade matrix X can be sparse since it is constructed using data from multiple areas and students only take a subset of courses. When X is constructed using all courses not only limited to the core courses, it can be highly sparse. The sparsity results in a difficult non-convex optimization problem which cannot be solved using standard SVD implementations. In this paper, we leverage the probabilistic matrix factorization method proposed in [20] to deal with the sparse matrix X, which is detailed as follows.

Probabilistic Matrix Factorization. We define the conditional distribution over the observed course grades as $p(X|U,V,\sigma^2) = \prod_{i=1}^{I}\prod_{j=1}^{J}[\mathcal{N}(X_{ij}|U_i^T V_j, \sigma^2)]^{\mathbf{1}_{ij}}$, where $\mathcal{N}(x|u,\sigma^2)$ is the probability density function of the Gaussian distribution with mean μ and variance σ^2, and $\mathbf{1}_{ij}$ is the indicator function that is equal to 1 if student i takes course j and equal to 0 otherwise. We also place zero-mean spherical Gaussian priors on student and course feature vectors:

$$p(U|\sigma_U^2) = \prod_{i=1}^{I}\mathcal{N}(U_i|0,\sigma_U^2\mathbf{I}), p(V|\sigma_V^2) = \prod_{j=1}^{J}\mathcal{N}(V_j|0,\sigma_V^2\mathbf{I})$$

The log of the posterior distribution over the student and course features is given by

$$\begin{aligned}&\ln p(U,V|X,\sigma^2,\sigma_V^2,\sigma_U^2)\\ =&-\frac{1}{2\sigma^2}\sum_{i=1}^{I}\sum_{j=1}^{J}\mathbf{1}_{ij}(X_{ij}-U_i^T V_j)^2\\ &-\frac{1}{2\sigma_U^2}\sum_{i=1}^{I}U_i^T U_i-\frac{1}{2\sigma_V^2}\sum_{j=1}^{M}V_j^T V_j \qquad (2)\\ &-\frac{1}{2}\left(\left(\sum_{i=1}^{I}\sum_{j=1}^{J}\mathbf{1}_{ij}\right)\ln\sigma^2+IK\ln\sigma_U^2+JK\ln\sigma_V^2\right)+C\end{aligned}$$

where C is a constant that does not depend on the parameters. Fix $\sigma^2, \sigma_U^2, \sigma_V^2$, maximizing the log-posterior over U and V is equivalent to minimizing the following regularized problem

$$\min_{U,V} \; \frac{1}{2}\sum_{i=1}^{I}\sum_{j=1}^{J}\mathbf{1}_{ij}(X_{ij}-U_i^T V_j)+\frac{\lambda_U}{2}\|U\|_F^2+\frac{\lambda_V}{2}\|V\|_F^2$$

where $\lambda_U = \frac{\sigma^2}{\sigma_U^2}$, $\lambda_V = \frac{\sigma^2}{\sigma_V^2}$ and $\|\cdot\|_F$ denotes the Frobenius norm. A local minimum of the objective function can be found by performing gradient descent in U and V. This model can be viewed as a probabilistic extension of the SVD model. In particular, if all course grades have been observed, the objective reduces to the SVD objective in the limit of prior variances going to infinity.

In order to estimate the best values of the parameters $\Omega \triangleq (\sigma^2, \sigma_U^2, \sigma_V^2)$, we utilize the Expectation-Maximization (EM) algorithm. The objective is to find the maximum likelihood estimate of Ω such that

$$\hat{\Omega} = \arg\max_{\Omega} p(X|\Omega) = \arg\max_{\Omega}\int p(X,U,V|\Omega)dUdV$$

The EM algorithm works iteratively. In the r-th iteration, the EM algorithm performs two steps: **Expectation Step**. Compute $g(\Omega) = \mathbb{E}_{(U,V|X,\hat{\Omega}^{(r-1)})}[\ln p(U,V|X,\Omega)]$, where $\ln p(U,V|X,\Omega)$ is given in (2) and $\hat{\Omega}^{(r-1)}$ is the estimate from the last iteration. Since the expectation is not in closed form, we draw Gibbs samples and compute the Monte Carlo mean. **Maximization Step**. Find $\hat{\Omega}^{(r)} = \arg\max_{\Omega} g(\Omega)$.

Once U and V are found through probabilistic matrix factorization, course clustering can then be performed. There are several ways to perform the clustering, below we describe two. **Method 1**: Each course j is assigned to a single cluster k in which the value of the cluster-course matrix V has the highest value among all possible K course clusters. Let $k(j)$ denote the cluster that course j belongs to, then $k(j) =$

$\arg\max_k V_{k,j}$. **Method 2**: Course clusters are determined using a thresholding rule. Specifically, course j belongs to cluster k if $V_{k,j} > \bar{v}$ where $\bar{v}$ is a predefined threshold value. In this case, a course may belong to multiple clusters. Our simulations adopt Method 1 to perform course clustering.

C. Training Base Predictors

Given the course clustering results, the base predictors are constructed by implementing off-the-shelf machine learning algorithms, such as linear regression, logistic regression, support vector machines, artificial neural networks etc. As we mentioned before, even for the same predictor, multiple instances are trained, one for each term. In term t, students in the area under consideration have taken courses $\bar{\mathcal{S}}^t$ following the recommended course sequence. Let $\mathcal{N}_{k(j^*)}$ be the course cluster to which the targeted course belongs to. For term t base predictor h^t, the feature vector $\tilde{\boldsymbol{x}}_i^t$ has a size equal to $|\bar{\mathcal{S}}^t \cap \mathcal{N}_{k(j^*)}|$. That is, only relevant courses that have been taken by term t are used in the training for h^t. If a student did not take a course in $\bar{\mathcal{S}}^t \cap \mathcal{N}_{k(j^*)}$ (e.g. the student is in a different area), then the corresponding entry is set to NULL. Clearly $\tilde{\boldsymbol{x}}_i^t$ is also expanding over t since $\bar{\mathcal{S}}^t$ is expanding. Given the new dataset $\{\tilde{\boldsymbol{x}}_i^t, y_i\}_{i\in\mathcal{I}}$, h^t can be trained by using any of the off-the-shelf algorithms.

V. Online Learning Ensemble Predictors

So far we have obtained a set of base predictors $\mathcal{H}^t$ for each term t. Let $H = |\mathcal{H}^t|$ denote the number of base predictors per term. The next question is how to make predictions using these base predictors for new students. There would not be any problem if there were just one term, i.e. $T = 1$ and one base predictor, i.e. $H = 1$. However, multiple base predictors and multiple terms introduce new challenges of (1) how to synthesize the prediction outcome of base predictors in each term and (2) how to incorporate the prediction outcome from previous terms. The second challenge is of particular interest since it is closely related to incorporating students' evolving performance. In this section, we propose an online progressive prediction architecture based on ensemble learning techniques.

A. Problem Formulation

To formalize the problem, we consider a stochastic setting where new students arrive in sequence $i = 1, 2, ...$ Such a stochastic setting is in fact suitable for both offline training and online updating. Given an offline training dataset, the student arrival sequence can be easily generated according to a random process. In the online scenario, the ensemble predictors are able to keep improving themselves using the new student data. Each student belongs to a student group depending on θ_i the static feature of the student (e.g. high school GPAs and SATs). The student group can be created by a variety of state-of-the-art clustering algorithms. Let $g_i \in \mathcal{G}$ be the group of student i and $\mathcal{G}$ be the group space. For instance, $\mathcal{G}$ can consist of a high SAT score student group and a low SAT score student group.

In each term t, each of the base predictor $h^t \in \mathcal{H}^t$ makes a prediction $z_{h,i}^t = h^t(\theta_i, \tilde{\boldsymbol{x}}_i^t)$ of the grade of a targeted course j^* that student i is supposed to take in a later term using the static feature θ_i as well as the performance vector $\tilde{\boldsymbol{x}}_i^t$ restricted to the relevant courses. Therefore, in term t, we have a total number of $t \times H$ prediction results per targeted course, namely $z_{h,i}^\tau, \forall \tau \leq t$. Our task is to synthesize these base prediction results and output a final prediction $\hat{y}_i^t$. The reason why we want to utilize all these base prediction results is that even though one particular base predictor performs the best in the offline testing, it may not be the best in the online setting since the underlying distribution of new students may not be exactly the same as that of students in the existing dataset. Sticking to one particular base predictor may end up with a very suboptimal performance. Note that base predictor h^t may not necessarily perform better than h^{t-1} since it is possible that the newly added course grade information represents noise.

Before presenting our algorithm, we define a performance measure which will be useful for the description and the analysis of our algorithm. The cumulative loss for a term t base predictor $h \in \mathcal{H}^t$ with respect to group g up to student n is defined as $L_n(h|g) = \sum_{i=1}^n l(z_{i,h}^t, y_i)\mathbf{1}\{g_i = g\}$, where $l(y', y)$ is a loss function that characterizes the prediction error. For instance, $l(y', y) = (y - y')^2$ for the regression case and $l(y', y) = \mathbf{1}\{y \neq y'\}$ for the classification case. The overall cumulative loss for a term t base predictor h up to student n is thus $L_n(h) = \sum_{g\in\mathcal{G}} L_n(h|g)$. These loss functions can be easily computed using the individual prediction of the base predictors and the realized performance of student 1 through n. Let $h^*(t|g) = \arg\min_{h\in\mathcal{H}^\tau, \forall\tau\leq t} L_n(h|g)$ be the best base predictor for group g among all base predictors up to term t in hindsight and $L_n^{*,t}(g) = L_n(h^*(t|g)|g)$ be the corresponding minimal cumulative loss.

Similarly, we define the cumulative loss for a term t ensemble predictor f^t with respect to group g of student 1 through n as $L_n(f^t|g) = \sum_{i=1}^n l(\hat{y}_i^t, y_i)\mathbf{1}\{g_i = g\}$ and the overall cumulative loss as $L_n(f^t) = \sum_{g\in\mathcal{G}} L_n(f^t|g)$. These cumulative loss functions will depend on how the ensemble prediction is made based on the base prediction results. Our objective is to synthesize the local prediction results such that the long-term performance of the ensemble predictor is at least as good as that of the optimal individual base predictor in hindsight.

B. Progressive Prediction

A major challenge in utilizing all the base prediction results $z_{h,i}^t, \forall\tau \leq t$ is that the number of these prediction results grows linearly with the number of terms and can be very large when t is large. Moreover, even though the previous-term predictors $h^\tau, \tau < t$ may still add valuable information, in general their predictive power is less than the current-term predictor h^t and hence should be treated differently. To address these two issues, we propose a progressive prediction architecture based on ensemble learning techniques. Specifically, for each term t we construct an ensemble predictor f^t. The input to f^t is the prediction results of the term t base predictors $\mathcal{H}^t$ and the ensemble prediction from the previous term. Due to the cascading structure, the prediction results of all base predictors up to term t is implicitly taken into account.

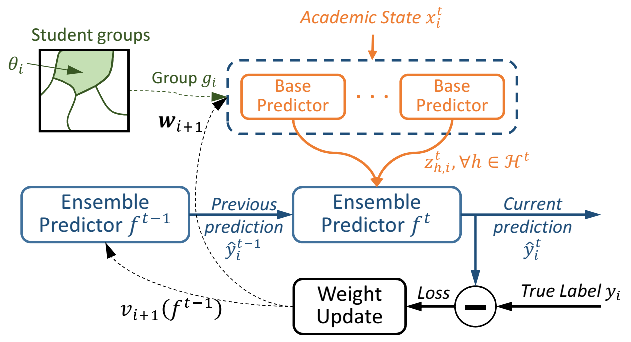


Fig. 4. Ensemble Prediction and Weight Update.

The term t ensemble predictor is learned using the exponentially weighted average forecaster (EWAF) algorithm. Specifically, each base predictor h^t is associated with a weight vector $\boldsymbol{w}_i(h^t)$ and the previous term ensemble predictor f^{t-1} is associated with a weight vector $\boldsymbol{v}_i(f^{t-1})$. All these weight vectors have a size $|\mathcal{G}|$ equal to the number of student groups. Therefore, weights are maintained for each student group. The weights are updated when the student performance in the targeted course j^* is realized and hence are changing over the student index i. All weight vectors are initialized to be $(1, ..., 1)$. For student i, at the beginning of term t, the ensemble predictor f^t makes a prediction $\hat{y}_i^t$ based on a weighted average of the base prediction results $z_{h,i}^t, \forall h \in \mathcal{H}^t$ and the previous-term ensemble prediction results $\hat{y}_i^{t-1}$. Depending on whether the prediction is a regression problem or a classification problem, the weighted averaging is deterministic or probabilistic as follows.

Regression. The ensemble prediction is the weighted average of the input predictions:

$$\hat{y}_i^t = \frac{\sum_{h\in\mathcal{H}^t} w_{i,g}(h) z_{i,h}^t + v_{i,g}(f^{t-1})\hat{y}_i^{t-1}}{\sum_{h\in\mathcal{H}^t} w_{i,g}(h) + v_{i,g}(f^{t-1})} \tag{3}$$

Classification. The ensemble prediction is $\hat{y}_i^{t-1}$ (or $z_{i,h}^t$) with probability

$$\frac{v_{i,g}(f^{t-1})}{\sum_{h\in\mathcal{H}^t} w_{i,g}(h) + v_{i,g}(f^{t-1})} (\text{or} \frac{w_{i,g}(h)}{\sum_{h\in\mathcal{H}^t} w_{i,g}(h) + v_{i,g}(f^{t-1})})$$

Using the later realized performance in the targeted course j^* of student i, the weights of the base predictors and the previous-term ensemble predictor are updated according to their cumulative loss for group g students. Specifically, the weights to be applied to student $i+1$ are updated to

$$w_{i+1,g}^t(h^t) = \exp(-\eta_i L_i(h^t|g)) \tag{4}$$
$$v_{i+1,g}^{t-1}(h^t) = \exp(-\eta_i L_i(f^{t-1}|g)), \quad \text{if } \ g = g_i \tag{5}$$

where η_i is a sequence of input constants. For $g \neq g_i$, the weights remain the same. Intuitively, the predictor with a larger cumulative loss will be assigned with a smaller weight and hence its prediction for future students will be considered less important in the ensemble synthesis. Figure 4 illustrates the ensemble prediction and the weight update. Algorithm 1 provides the pseudo code of the proposed Ensemble-based Progressive Prediction (EPP) algorithm.

Algorithm 1 Ensemble-based Progressive Prediction (EPP)

```
1:  Initialization: $L(h^t) = 0, L(f^t) = 0, \forall t$.
2:  for each student $i$ do
3:      Observe backgrounds $\theta_i$, student group $g_i$
4:      for term $t = 1$ to $T$ do                          ▷ Prediction Phase
5:          Observe performance state $\boldsymbol{x}_i^t$
6:          Extract relevant state $\tilde{x}_i^t$
7:          Receive prediction $\hat{y}_i^{t-1}$ from $f^{t-1}$
8:          Base predictor $h \in \mathcal{H}^t$ predicts $z_{h,i}^t = h^t(\theta_i, \tilde{x}_i^t)$
9:          Ensemble predictor $f^t$ predicts
10:             $\hat{y}_i^t = f^t(\hat{y}_i^{t-1}, \{z_{h,i}^t\}_h | \boldsymbol{v}_i^{t-1}, \boldsymbol{w}_i^t)$
11:     end for
12:     Observe true label $y_i$.
13:     for term $t = 1$ to $T$ do                          ▷ Update Phase
14:         Compute prediction loss $l(\hat{y}_i^t, y_i)$ and $l(z_{i,h^t}^t, y_i)$
15:         Update $L_i(h^t|g_i) \leftarrow L_{i-1}(h^t|g_i) + l(z_{i,h^t}^t, y_i)$
16:             $L_i(f^{t-1}|g_i) \leftarrow L_{i-1}(f^{t-1}|g_i) + l(\hat{y}_i^t, y_i)$
17:         Update weights $\boldsymbol{w}_{i+1}^t$ and $\boldsymbol{v}_{i+1}^t$ according to (4)
18:     end for
19: end for
```

C. Performance Analysis

In this section, we characterize the performance of the proposed progressive predictor. We study only the case $|\mathcal{G}| = 1$ since the extension is straightforward. We will focus on the ensemble predictor for a particular term t and compare it with the best base predictor among terms $1, 2, ..., t$ in hindsight. Once the performance of ensemble predictor f^t is characterized, characterizing the overall performance of the progressive predictor is also straightforward. However, since the ensemble prediction is deterministic in the regression case whereas it is probabilistic in the classification case, we analyze the performance separately for these two cases.

Regression Case. We define the regret of ensemble predictor f^t up to student n as the difference between its cumulative loss $L_n(f^t)$ and the cumulative loss of the best base predictor $L_n^{*,t}$, denoted by $\text{Reg}^t(n) = L_n(f^t) - L_n^{*,t}$.

Proposition 1 (Regret for Regression)**.** *When the EPP algorithm runs with parameter $\eta_i = \sqrt{8(\ln(H+1))/i}$, then for any number n of students, the regret of ensemble predictor f^t satisfies (1) $Reg^t(n) < O(\sqrt{n})$ and (2) $\mathbb{E}[Reg^t(n)] \leq O((t - \sum_{\tau=1}^t \rho^\tau \tau + 1)\sqrt{n \ln(H+1)})$ assuming that the best base predictor h^* is in term τ with probability ρ^τ (we must have $\sum_{\tau=1}^t \rho^\tau = 1$).*

Proof. This proof is largely based on Theorem 2.3 in [18], which can be establish regret bounds on the performance difference between f^t and the best among $\mathcal{H}^t$ when we restrict to the ensemble learning problem in term t. In particular, we have

$$\begin{aligned} &L_n(f^t) - \min\{L_n(f^{t-1}), L_n(h), \forall h \in \mathcal{H}^t\} \\ \leq &2\sqrt{\frac{n}{2}\ln(H+1)} + \sqrt{\frac{\ln(H+1)}{8}} \end{aligned} \tag{6}$$

Now consider term $t-1$, we have the same bound for f^{t-1}

$$\begin{aligned}&L_n(f^{t-1}) - \min\{L_n(f^{t-2}), L_n(h), \forall h \in \mathcal{H}^{t-1}\}\\ \leq &2\sqrt{\frac{n}{2}\ln(H+1)} + \sqrt{\frac{\ln(H+1)}{8}}\end{aligned} \tag{7}$$

Combining both bounds, we have

$$\begin{aligned}&L_n(f^t) - \min\{L_n(h), \forall h \in \mathcal{H}^{t-1}\}\\ =&L_n(f^t) - L_n(f^{t-1}) + L_n(f^{t-1}) - \min\{L_n(h), \forall h \in \mathcal{H}^{t-1}\}\\ =&\leq 2\left(2\sqrt{\frac{n}{2}\ln(H+1)} + \sqrt{\frac{\ln(H+1)}{8}}\right)\end{aligned} \tag{8}$$

By induction, we have $\forall \tau \leq t$

$$\begin{aligned}&L_n(f^t) - \min\{L_n(h), \forall h \in \mathcal{H}^{\tau}\}\\ \leq&(t-\tau+1)\left(2\sqrt{\frac{n}{2}\ln(H+1)} + \sqrt{\frac{\ln(H+1)}{8}}\right)\end{aligned} \tag{9}$$

Therefore,

$$\text{Reg}^t(n) \leq t\left(2\sqrt{\frac{n}{2}\ln(H+1)} + \sqrt{\frac{\ln(H+1)}{8}}\right) \tag{10}$$

and

$$\begin{aligned}&\mathbb{E}[\text{Reg}^t(n)] \qquad (11)\\ \leq&\mathbb{E}(t-\tau^*+1)\left(2\sqrt{\frac{n}{2}\ln(H+1)} + \sqrt{\frac{\ln(H+1)}{8}}\right)\\ \leq&(t-\sum_{\tau=1}^{t}\rho^\tau\tau+1)\left(2\sqrt{\frac{n}{2}\ln(H+1)} + \sqrt{\frac{\ln(H+1)}{8}}\right)\end{aligned}$$

where τ^* is the term of the optimal base predictor in hindsight. □

Several remarks are made as follows.

(1) The regret bound is $O(\sqrt{n})$, which is sublinear in the number of students n. This implies that the average regret tends to zero when $n \to \infty$, i.e. $\lim_{n\to\infty}\frac{1}{n}\text{Reg}^t(n) \to 0$. This bound provides a worst-case performance guarantee for the ensemble predictor, stating that the ensemble predictor is no worse than the best base predictor in hindsight asymptotically.

(2) Another way to perform the ensemble learning in term t is to learn directly among all the base predictors up to term t. Therefore, the input to f^t is $z^\tau_{h,i}, \forall h \in \mathcal{H}^\tau, \forall \tau \leq t$. Figure 5 illustrates the difference between the direct method and the proposed progressive prediction method. However, the direct method not only induces significantly larger complexity since the ensemble learning in each term becomes much more complicated but also may have worse expected regret performance. In particular, the expected regret bound for the direct method can be shown as $\mathbb{E}[\text{Reg}^t(n)] \leq O(\sqrt{n\ln(Ht)})$ which is independent of the probability distribution ρ^τ.

(3) Proposition 1 in fact highlights the importance of taking the student's evolving progress into account when predicting the student's future performance. Consider a particular term t. The base predictor $\mathcal{H}^t$ uses only the current performance state but not the previous performance state to make the prediction whereas the ensemble predictor f^t implicitly uses

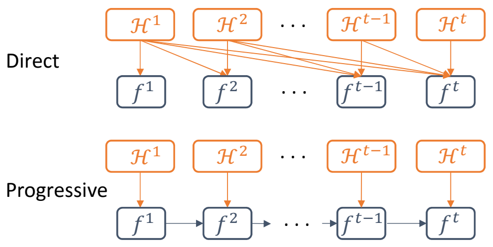


Fig. 5. Difference between direct method and progressive method.

all past performance states. The cumulative loss $L_n(h^t)$ clearly is no less than $L_n^{*,t}$ on average for any t. Since $L_n(f^t)$ is asymptotically no more than $L_n^{*,t}$ on average, it can be concluded that using the student's evolving progress improves the prediction performance.

Classification Case. Since the prediction output in the classification case is randomly sampled according to a distribution, we define regret in a slightly different way. Instead of using the realized cumulative loss, we use the expected cumulative loss when defining regret: $\text{Reg}^t(n) = \mathbb{E}[L_n(f^t)] - L_n^{*,t}$. Because of the probabilistic prediction, the space of predictions and the loss functions are not convex in the classification case and hence, results of Theorem 2.3 in [18] are no longer applicable. Our prior work [32] establishes a similar result for the classification case, which is utilized in the regret analysis in this paper.

Proposition 2 (Theorem 2 in [32])**.** *When the EPP algorithm runs with parameter $\eta_i = \sqrt{\ln(H+1)/i}$, then we have $\mathbf{E}[L_n(f^t)] - \min\{L_n(f^{t-1}), L_n(h), \forall h \in \mathcal{H}^t\} \leq 2\sqrt{n\ln(H+1)}$.*

Using Proposition 2, we can perform a similar analysis for the classification case and obtain the following proposition.

Proposition 3 (Regret for Classification)**.** *Assume that the best base predictor h^* is in term τ with probability ρ^τ (we must have $\sum_{\tau=1}^t \rho^\tau = 1$). When the EPP algorithm runs with parameter $\eta_i = \sqrt{\ln(H+1)/i}$, then for any number n of students, the regret of ensemble predictor f^t satisfies (1) $Reg^t(n) < O(\sqrt{n})$ and (2) $\mathbb{E}[Reg^t(n)] \leq O((t - \sum_{\tau=1}^t \rho^\tau\tau + 1)\sqrt{n\ln(H+1)})$.*

The implications of Proposition 3 for the classification case are similar to those of Proposition 1 for the regression case.

VI. Experimental Results

A. Dataset

Student data used to test our method is collected from the Mechanical and Aerospace Engineering department at UCLA across students who graduated in three years (2013, 2014, 2015). The dataset has 1169 anonymized undergraduate students enrolled in the program in two different areas (Mechanical Engineering and Aerospace Engineering). All students completed their program. We excluded the transferred students to UCLA since the course information of these students before the transfer is missing. The data of each student contains the student' pre-college traits (high school GPA and SAT scores),

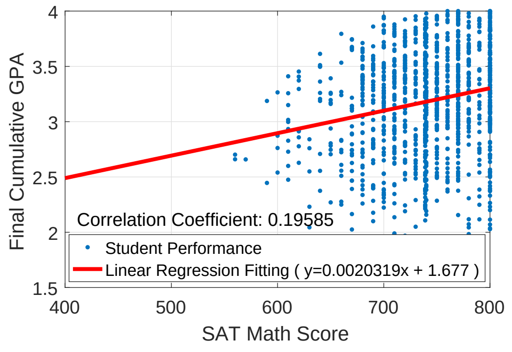


Fig. 6. Correlation between SAT Math and final GPA.

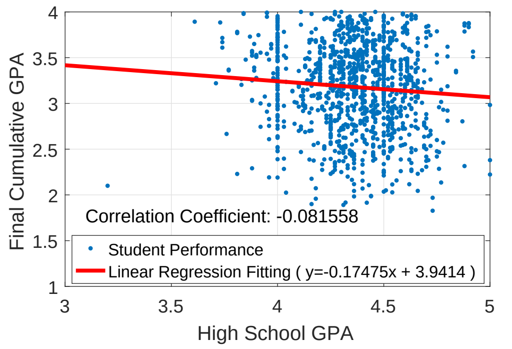


Fig. 7. Correlation between high school GPA and final GPA.

the courses (including lectures and labs) that the students take in each academic quarter, the course credits and the obtained grades. We use only high school GPA and SAT scores as the static features of students due to the limitation of the dataset. Nevertheless, other features, if available, can be used in our framework. UCLA adopts a quarter system and hence, each academic term is a quarter. We consider only the fall, winter and spring quarters but not the summer quarter which has a different enrollment pattern and different course durations (i.e. shorter than courses in other quarters). We consider only letter-grade courses but not pass/fail courses.

Figure 6 shows the correlation between the SAT math score of the students and their final GPA, and Figure 7 shows

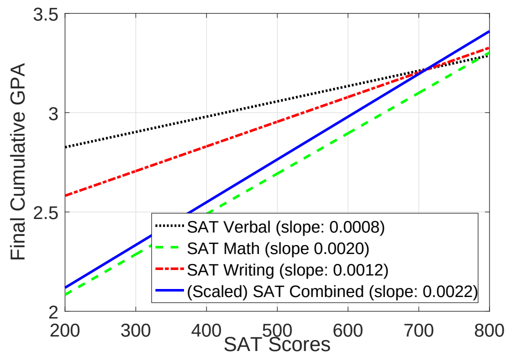


Fig. 8. Correlation between SAT Verbal/Math/Writing/Combined and final GPA

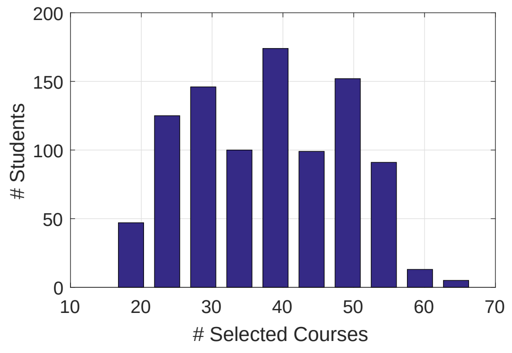


Fig. 9. Distribution of course selection

the correlation between the high school GPA and the final GPA. The general trend is that students with higher SAT scores also obtain higher final GPA in the degree program. However, high school GPA is almost not correlated with the final GPA in the college, suggesting that high school GPA has weaker predictive power than SAT scores. Figure 8 further illustrates the correlation (obtained by linear fitting) between SAT verbal/math/writing/combined and final GPA. SAT combined score has the highest correlation with the final GPA. SAT math is the most powerful predictor among individual SAT scores, which is reasonable since students are from an engineering department.

Figure 9 illustrates the distribution of the number of courses selected by students. The average number of courses that students take is 38. Although students take a similar number of courses, the courses that they take are extremely diverse. The total number of distinct courses among all students is 811 while 759 of them are taken by less than 10% of the students. The diversity is mainly due to the elective courses, which can be vastly different depending on students' own interest. We observed that a significantly large portion of the courses are taken by only one student in our dataset, making the distribution extremely biased towards the lower percentage end.

B. *Learning Correlated Courses*

We performed probabilistic matrix factorization on the student dataset to learn the course correlation. We explored different numbers of course clusters by setting $K = 5, 10, 20$. Figures 10 and 11 show the discovered V matrix for core courses in a colormap representation, where the values are represented by color, for $K = 20$ and $K = 5$, respectively.

Below we provide a couple of case studies to show how the clustering results relate to and are different from course clustering using educational domain knowledge.

MAE 182A. MAE 182A is the course "Mathematics of Engineering". The prerequisite courses of MAE 182A are shown in Figure 12 which are all math courses. Besides these courses, our methods discover several other correlated courses, including CHEM 20BH, EE 110L, MAE 102, MAE 105A and PHYS 1A, even though they are not prerequisite courses of MAE 182A. Table I reports the correlation coefficients obtained by using our matrix factorization method and the

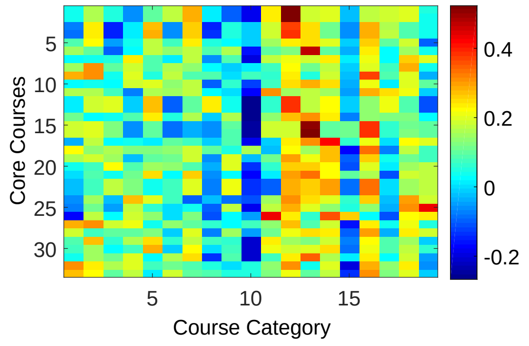


Fig. 10. Matrix Factorization Results (the V matrix restricted to the core courses) in Colormap ($K = 20$).

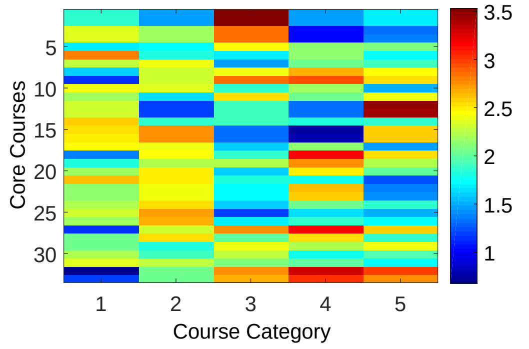


Fig. 11. Matrix Factorization Results (the V matrix restricted to the core courses) in Colormap ($K = 5$).

Pearson correlation coefficients for a number of courses. As we can see, our method is effective in discovering the correlation among courses. In particular,

EE 100 and EE 110L. These two courses are "Electrical and Electronics Circuits" and "Circuit Measurements Laboratory". Figure 13 shows their prerequisite courses. Besides the prerequisite courses, our method discovered three more courses that have predictive power, namely CHEM 20B/BH, MAE 103, MAE 102, and MAT SCI 104. However, our method shows that MATH 32B is not a strongly correlated course even though it is a co-requisite course of PHYS 1B and PHYS 1C.

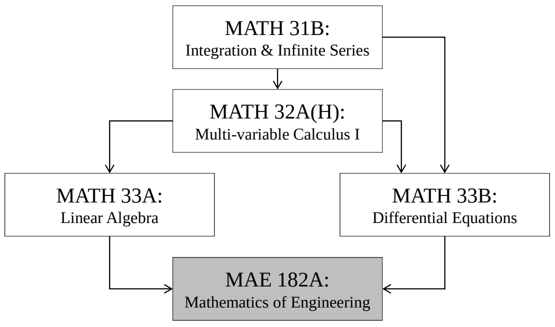


Fig. 12. Prerequisite courses of MAE 182A. (Dashed lines are co-requisite courses)

TABLE I
CORRELATION COEFFICIENTS AND THE SLOPE OF THE LINEAR FIT

Course Name	MF	Pearson	Note
MATH 33A	0.3103	0.3278	Direct Pre-requisite
MATH 33B	0.4668	0.4505	Direct Pre-requisite
MATH 31B	0.0924	0.0874	Indirect Pre-requisite
MATH 32A	0.2624	0.1986	Indirect Pre-requisite
MAE 105A	0.4862	0.4779	Additionally Discovered
CHEM 20B	0.4205	0.4072	Additionally Discovered
MAE 103	0.4205	0.3942	Additionally Discovered
MAT SCI 104	0.4096	0.4041	Additionally Discovered
MAE 105D	0.1775	0.1997	Excluded
MAE 94	0.1243	0.1053	Excluded

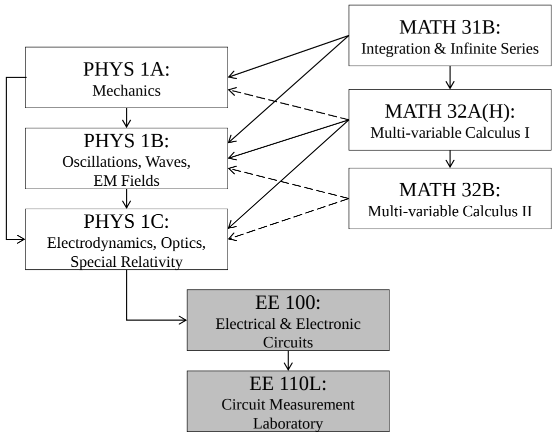

MATH 31B:
Integration & Infinite Series
PHYS 1A:
Mechanics
MATH 32A(H):
Multi-variable Calculus I
PHYS 1B:
Oscillations, Waves, EM Fields
MATH 32B:
Multi-variable Calculus II
PHYS 1C:
Electrodynamics, Optics, Special Relativity
EE 100:
Electrical & Electronic Circuits
EE 110L:
Circuit Measurement Laboratory

Fig. 13. Prerequisite courses of EE 100 and EE 110L.

C. Prediction Performance

We constructed four base predictors for each quarter implemented by four classic machine learning algorithms: linear regression, logistic regression, random forest and k-Nearest Neighbors (kNN). Base predictors in quarter t are trained using course grade data up to quarter t restricted to the discovered relevant courses of the targeted course.

We point out an important difference between the setup used in the simulation and that in the model. In our model, we assumed that students in the same area follow the exact same course sequence and take the exact (core) courses in each quarter. In practice, however, students may still take courses in slightly different orders. As a result, some students may take a targeted course later than the recommended quarter in which the course should be taken. Therefore, in the simulations, the prediction performance for a targeted course is shown for the entire program duration, namely 12 quarters.

Figures 14 and 15 show the mean square errors of the grade prediction for MAE 182A and EE 110L, respectively. In general, the prediction error decreases over the quarters for all the base predictors as well as the ensemble predictor. This is because more information is accumulated over time. Note that the first data point in all curves correspond to a model that uses only static matriculating measurements but not the evolving performance of students. Among the base predictors that we implemented, random forest performs the best, kNN performs the worst in most cases. The proposed progressive prediction algorithm outperforms all base predictors since it not only

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2017.2692560, IEEE Journal of Selected Topics in Signal Processing

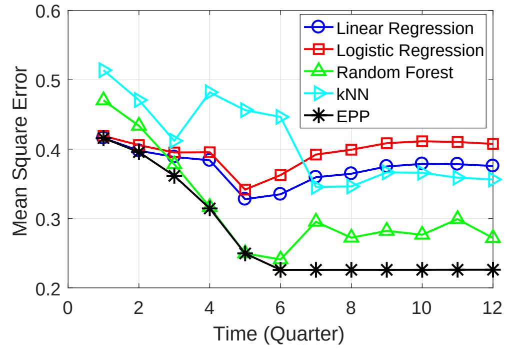


Fig. 14. Prediction performance for MAE 182A. (Ensemble v.s. Base)

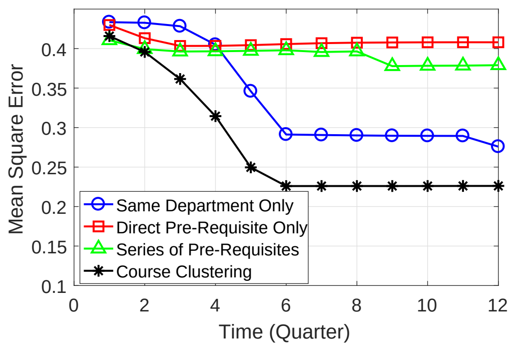


Fig. 16. Prediction performance for MAE 182A. (Different Inputs)

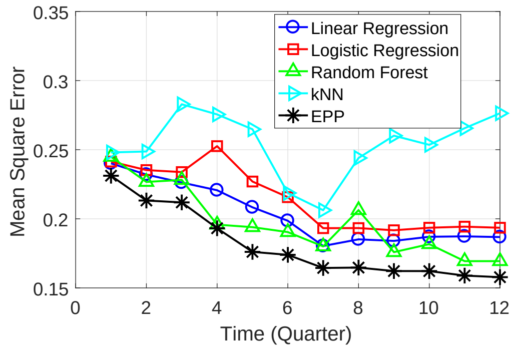


Fig. 15. Prediction performance for EE 110L. (Ensemble v.s. Base)

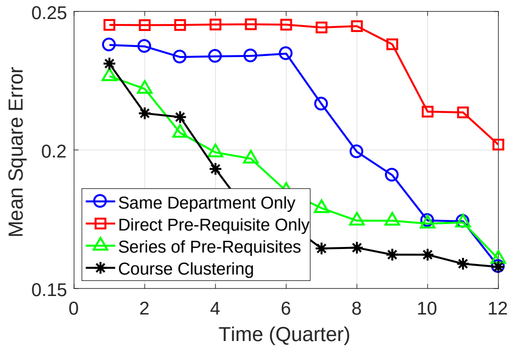


Fig. 17. Prediction performance for EE 110L. (Different Inputs)

utilizes the prediction of the base predictors in the current quarter but also utilizes the previous quarter prediction results. We also perform the experiments on three other courses EE100, MAE105D and PHYSICS4BL in all of which the proposed ENN algorithm beats the base predictors.

We further compare the prediction performance with algorithms taking different input features in order to show the impact of learning relevant courses. Specifically, we compare three benchmarks based on the educational domain knowledge: **Same Department Only**. The courses in a curriculum are offered by different departments. For instance, in the MAE curriculum, courses EE 100, EE 110L etc. are offered by Electrical Engineering department. Courses MATH 33A, MATH 33B etc. are offered by Mathematics department. In this benchmark, only courses offered by the same department as that of the targeted course are considered relevant; **Direct Prerequisite Only** In this benchmark, only the direct pre-requisite courses are considered as the relevant courses of the targeted course. For instance, the direct pre-requisite courses of MAE 182A are MATH 33A and MATH 33B; **Series of Prerequisites**. Direct prerequisite courses may also have their pre-requisite courses. In this benchmark, the series of the prerequisite courses are considered as the relevant courses of the targeted course. For instance, for MAE 182A, in addition to MATH 33A and MATH 33B, MATH 32A and MATH 31B are also considered as the input to the prediction algorithm.

Figures 16 and 17 compare the prediction performance. For MAE 182A, courses offered by the same department have more predictive power than the prerequisite courses. For EE 110L, prerequisite courses have more predictive power than courses offered by the same department. In both cases, our algorithm based on course clustering yields the best prediction performance.

VII. Conclusions

In this paper, we proposed a novel method for predicting students' future performance in degree programs given their current and past performance. A latent factor model-based course clustering method was developed to discover relevant courses for constructing base predictors. An ensemble-based progressive prediction architecture was developed to incorporate students' evolving performance into the prediction. These data-driven methods can be used in conjunction with other pedagogical methods for evaluating students' performance and provide valuable information for academic advisors to recommend subsequent courses to students and carry out pedagogical intervention measures if necessary. Additionally, this work will also impact curriculum design in degree programs and education policy design in general. Future work includes extending the performance prediction to elective courses and using the prediction results to recommend courses to students.

References

[1] The White House, "Making college affordable," https://www.whitehouse.gov/issues/education/higher-education/making-college-affordable, 2016.

[2] Complete College America, "Four-year myth: Making college more affordable," http://completecollege.org/wp-content/uploads/2014/11/4-Year-Myth.pdf, 2014.

[3] H. Cen, K. Koedinger, and B. Junker, "Learning factors analysis–a general method for cognitive model evaluation and improvement," in *International Conference on Intelligent Tutoring Systems*. Springer, 2006, pp. 164–175.

[4] M. Feng, N. Heffernan, and K. Koedinger, "Addressing the assessment challenge with an online system that tutors as it assesses," *User Modeling and User-Adapted Interaction*, vol. 19, no. 3, pp. 243–266, 2009.

[5] H.-F. Yu, H.-Y. Lo, H.-P. Hsieh, J.-K. Lou, T. G. McKenzie, J.-W. Chou, P.-H. Chung, C.-H. Ho, C.-F. Chang, Y.-H. Wei *et al.*, "Feature engineering and classifier ensemble for kdd cup 2010," in *Proceedings of the KDD Cup 2010 Workshop*, 2010, pp. 1–16.

[6] Z. A. Pardos and N. T. Heffernan, "Using hmms and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset," *Journal of Machine Learning Research W & CP*, 2010.

[7] Y. Meier, J. Xu, O. Atan, and M. van der Schaar, "Personalized grade prediction: A data mining approach," in *Data Mining (ICDM), 2015 IEEE International Conference on*. IEEE, 2015, pp. 907–912.

[8] C. G. Brinton and M. Chiang, "Mooc performance prediction via clickstream data and social learning networks," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 2299–2307.

[9] KDD Cup, "Educational data minding challenge," https://pslcdatashop.web.cmu.edu/KDDCup/, 2010.

[10] Y. Jiang, R. S. Baker, L. Paquette, M. San Pedro, and N. T. Heffernan, "Learning, moment-by-moment and over the long term," in *International Conference on Artificial Intelligence in Education*. Springer, 2015, pp. 654–657.

[11] C. Marquez-Vera, C. Romero, and S. Ventura, "Predicting school failure using data mining," in *Educational Data Mining 2011*, 2010.

[12] Y.-h. Wang and H.-C. Liao, "Data mining for adaptive learning in a tesl-based e-learning system," *Expert Systems with Applications*, vol. 38, no. 6, pp. 6480–6485, 2011.

[13] N. Thai-Nghe, L. Drumond, T. Horváth, L. Schmidt-Thieme *et al.*, "Multi-relational factorization models for predicting student performance," in *Proc. of the KDD Workshop on Knowledge Discovery in Educational Data*. Citeseer, 2011.

[14] A. Toscher and M. Jahrer, "Collaborative filtering applied to educational data mining," *KDD cup*, 2010.

[15] R. Bekele and W. Menzel, "A bayesian approach to predict performance of a student (bapps): A case with ethiopian students," *algorithms*, vol. 22, no. 23, p. 24, 2005.

[16] N. Thai-Nghe, T. Horváth, and L. Schmidt-Thieme, "Factorization models for forecasting student performance," in *Educational Data Mining 2011*, 2010.

[17] Y. Meier, J. Xu, O. Atan, and M. van der Schaar, "Predicting grades," *IEEE Transactions on Signal Processing*, vol. 64, no. 4, pp. 959–972, Feb 2016.

[18] N. Cesa-Bianchi and G. Lugosi, *Prediction, learning, and games*. Cambridge university press, 2006.

[19] Y. Koren, R. Bell, C. Volinsky *et al.*, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[20] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *NIPS*, vol. 20, 2011, pp. 1–8.

[21] M.-C. Yuen, I. King, and K.-S. Leung, "Task recommendation in crowdsourcing systems," in *Proceedings of the First International Workshop on Crowdsourcing and Data Mining*. ACM, 2012, pp. 22–26.

[22] K. Christakopoulou and A. Banerjee, "Collaborative ranking with a push at the top," in *Proceedings of the 24th International Conference on World Wide Web*. ACM, 2015, pp. 205–215.

[23] Y. Xu, Z. Chen, J. Yin, Z. Wu, and T. Yao, "Learning to recommend with user generated content," in *International Conference on Web-Age Information Management*. Springer, 2015, pp. 221–232.

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

[25] N. Thai-Nghe, L. Drumond, A. Krohn-Grimberghe, and L. Schmidt-Thieme, "Recommender system for predicting student performance," *Procedia Computer Science*, vol. 1, no. 2, pp. 2811–2819, 2010.

[26] J. I. Lee and E. Brunskill, "The impact on individualizing student models on necessary practice opportunities." *International Educational Data Mining Society*, 2012.

[27] T. Mandel, Y.-E. Liu, S. Levine, E. Brunskill, and Z. Popovic, "Offline policy evaluation across representations with applications to educational games," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, 2014, pp. 1077–1084.

[28] E. Brunskill and S. Russell, "Partially observable sequential decision making for problem selection in an intelligent tutoring system," in *Educational Data Mining 2011*, 2010.

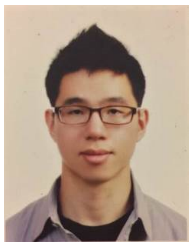
[29] J. Xu, T. Xing, and M. van der Schaar, "Personalized course sequence recommendations," *IEEE Transactions on Signal Processing*, vol. 64, no. 20, pp. 5340–5352, Oct 2016.

[30] W. Hoiles and M. van der Schaar, "Bounded off-policy evaluation with missing data for course recommendation and curriculum design," in *Proceedings of The 33rd International Conference on Machine Learning*, 2016, pp. 1596–1604.

[31] M. Cucuringu, C. Marshak, D. Montag, and P. Rombach, "Rank aggregation for course sequence discovery," *arXiv preprint arXiv:1603.02695*, 2016.

[32] C. Tekin, J. Yoon, and M. van der Schaar, "Adaptive ensemble learning with confidence bounds," *arXiv preprint arXiv:1512.07446*, 2015.

Jie Xu is an Assistant Professor in Electrical and Computer Engineering Department at the University of Miami. He received the B.S. and M.S. degrees in Electronic Engineering from Tsinghua University, Beijing, China, in 2008 and 2010, respectively and the Ph.D. degree in Electrical Engineering from UCLA. His primary research interests include game theory, multiagent systems, machine learning and edge computing.



Kyeong Ho Moon received B.Sc. degree in electrical and computer engineering from Cornell University, New York, in 2015. He is currently pursuing M.Sc. degree and Ph.D. degree in electrical engineering at University of California, Los Angeles. His research interests include machine learning, deep learning, and recurrent neural networks and their applications to medicine and education.

Mihaela van der Schaar is Chancellor's Professor of Electrical Engineering at University of California, Los Angeles. She is also Man Professor of Quantitative Finance in the Oxford-Man Institute of Quantitative Finance (OMI) and the Department of Engineering Science at Oxford, Fellow of Christ Church College and Faculty Fellow of the Alan Turing Institute, London. She is an IEEE Fellow, was a Distinguished Lecturer of the Communications Society (2011-2012), the Editor in Chief of IEEE Transactions on Multimedia (2011-2013). Her research interests include engineering economics and game theory, multiagent learning, online learning, decision theory, network science, multi-user networking, Big data and real-time stream mining, and multimedia.