Data-Driven Approach Towards a Personalized Curriculum

Michael Backenköhler, Felix Scherzinger, Adish Singla and Verena Wolf

June 29, 2018

Data-Driven Approach Towards a Personalized Curriculum

Michael Backenköhler
Saarland University
michael.backenkoehler
@uni-saarland.de

Felix Scherzinger
Saarland University

Adish Singla
MPI-SWS
adishs@mpi-sws.org

Verena Wolf
Saarland University
wolf@cs.uni-saarland.de

ABSTRACT

Course selection can be a daunting task, especially for first-year students. Sub-optimal selection can lead to bad performance of students and increase the dropout rate. Given the availability of historic data about student performances, it is possible to aid students in the selection of appropriate courses. Here, we propose a method to compose a personalized curriculum for a given student. We develop a modular approach that combines a context-aware grade prediction with statistical information on the useful temporal ordering of courses. This allows for meaningful course recommendations, both for fresh and senior students. We demonstrate the approach using the data of the computer science Bachelor students at Saarland University.

1. INTRODUCTION

Students at higher education institutions usually have to choose from a large set of possible courses in order to achieve an academic degree. Even for senior students, it is not obvious which courses to follow and in what sequence as the number of possible choices is large. Students often have problems to ensure progress in a program of study, especially in the first years of study, and to graduate in a timely manner.

Student success is also an important objective for decision makers at universities, which continuously monitor dropout rates and average times to degree. Completion rates at European universities range between 39% to 85% and are highly program dependent, while the average time-to-degree is around 3.5 years for a Bachelor degree [17].

When pursuing a degree students typically have to complete a set of mandatory courses, as well as courses that can be chosen more freely. In the first years, an adequate order of mandatory courses is of interest while in later years the focus is on the question which courses to take in general and which not. Instead of relying on individual recommendations from other students, our goal is to take advantage of the combined experience of former students and address both, an adequate temporal ordering and an intelligent selection of courses.

We propose an approach that combines statistical methods based on course orderings and grade prediction based on a collaborative filtering approach. This results in a model consisting of two main components, a course dependency graph and grade prediction. Therefore our model combines two major criteria: The expected performance, i.e. the expected grade, and preparedness, i.e. how prior course choices may benefit the student, for a given course.We believe that weaving the two criteria strongly increases the usability of our recommendations compared to previous work focusing only on one of the two.

To train our model we use long-term educational data of computer science Bachelor students from Saarland University's computer science department. The data consists of course performance information from several thousand students of various countries during the last ten years. Experiments with a first subset of students already showed promising results giving recommendations for first-year as well as for senior students.

2. RELATED WORK

Many course recommendation approaches are based on *performance prediction*. A wide range of standard machine learning methods have been applied to this problem [14, 15], as well as recommender system techniques [10]. Ray and Sharma [8] apply collaborative filtering based on item-item similarity. Ren et al. [9] supplement a matrix factorization approach with weights for recently taken courses. Besides a gain in predictive quality, the resulting model carries valuable information on beneficial orderings of courses. Polyzou and Kyrapis [7] propose a matrix factorization based on course-specific features. Slim et al. [12] use Markov networks of courses to predict individual grades and estimate the future performances inside a study program.

In contrast to the aforementioned approaches, our technique separates the concerns of performance and preparedness. This has the benefit of allowing for a custom weighting of the two components, as well as the increased explanatory value of the model itself.

Much effort on curriculum planning has been focused on

Massive Open Online Courses (MOOC). For instance, Hansen et al. [5] analyse characteristic question sequences in online courses by applying Markov chains to student clusters. Chen et al. [3] propose a squencing for items in the context of web-based courses.

In the context of university eduction, much effort has been directed towards providing analytical tools to educators and institutions. For example Zimmermann et al. [18] predict graduate performance, based on the students' undergraduate performances. Saarela and Kärkkäinen [11] analyse undergraduate student data to indentify relevant factors for a successful computer science education.

3. PROBLEM SETTING

We consider the problem of designing a student's curriculum that optimizes performance (measured in terms of course grades) and the time to degree. Hence, for each semester a subset of the courses offered is chosen such that the student's complete trace from the first semester until the final degree is (approximately) optimal, i.e., the performance and time to degree does not improve if the order in which the courses are taken is changed or if different courses are taken. We assume that a large number of traces of former students are given, including the particular grades achieved in each course. Note that this also includes data of students retaking courses are failing.However, the data may not provide information about students that enroll in a course but withdraw before the final exam. In addition, we assume that recommendations for students that already participated in certain courses, the corresponding partial trace is available as well as meta data about the student. Moreover, we want to take into account all selection rules of the corresponding study program.

The data-set consists of performance and meta-information of the students at the computer science department of Saarland University since 2006. It includes grades, basic information regarding students (age, nationality, sex, course of studies) as well as basic information regarding the lecture (course type, lecturer). Here, we consider a subset of 72 recurring courses which have a total of 16,090 entries of 1,700 students. A challenge regarding this particular data set is the fact that students may register fairly late in the semester for a particular course. Therefore the data does not capture a early student drop out.

4. COMPONENTS OF OUR APPROACH

In the context of standard recommender systems, the predicted rating is the basis for a recommendation. However, in the context of course recommendation, further aspects, such as the knowledge gain and constraints of the study program have to be taken into account. Here, we present an approach that is flexible enough to also incorporate such criteria in a modular way. Moreover, in our approach selection criteria can further be prioritized by the student. A student may, for example, prioritize taking a course that increases the preparedness for certain other courses. In this case, the course may be recommended although the students performance alone did not lead to suggestion of that course.

We construct a personalized *recommendation graph* of courses for each student based on the two main components: the course dependency graph and the performance prediction. The course dependency graph aims to capture the positive effect that course A has on the performance in course B. The performance prediction is done using a *collaborative filtering* approach, that incorporates contextual features of both the student and the course.

4.1 Course Dependency Graph

The Course Dependency Graph is a graph whose node set equals the set of all (regularly or irregularly offered) courses. A directed edge between course A and course B means that when passing A before B then the chance of getting a better grade in B is higher compared to the grade in B obtained for the order B before A.

We use the *Mann-Whitney U-test* [2] to construct such a graph of courses. The hypothesis of the test is that one random variable is smaller than another. If we let the random variable $X_{<c}$ denote the grade in course B for a student that had a grade $< c$ in course A an edge represents the hypothesis

$$\Pr(X_{<c} < k) > \Pr(X_{\geq c} < k),$$

where $X_{\geq c}$ includes the case of not taking course A. The hypothesis describes that the probability of drawing a grade of subset $X_{<c}$ which is better than k is higher than doing the same for subset $X_{\geq c}$.We fix a small significance level $\alpha = 0.0001$, to find the most important course relations. Since the test is quite sensitive, it tends to identify too many course pairs for higher significance levels. Moreover, a minimum number of 20 samples is required for each case to perform the test. The graph only contains an edge between two courses if the test confirms the above hypothesis.

In Germany grades are numbers in the set

$$P = \{1, 1.3, 1.7, 2, 2.3, 2.7, 3, 3.3, 3.7, 4, 5\},$$

where lower numbers are better and 5 is the failing grade. In general, we assume these performances to be normalized to mean zero and unit variance w.r.t. courses.

To construct the course dependency graph, we first construct one graph for each grade threshold $c \in P$. Next we average over the edges of all graphs, resulting in edge weights between 0 and 1. In this way the final graph, in which course dependency is not binary but a weighting, is more informative. A large value implies that this course ordering is beneficial to students of all performance levels while a low value indicates that this ordering is only helpful for a smaller set of students. Note that the absence of edges indicates that there is not enough information about the relation between the two courses.

An excerpt of a course dependency graph is shown in Figure 1. We find that 'Programming I', 'Maths I' and 'Maths II' are good starting points in this graph for a first-year student as they do not have incoming edges. Note that the missing edge between 'Maths I' and 'Maths II' is meaningful as 'Maths I' focuses on Linear Algebra while 'Maths II' is concerned with Analysis. As opposed to this, for 'Programming I' and 'II' the graph suggests to first take 'Programming I' as a preparation, which is a meaningful recommendation as the contents of 'Programming II' are based on those of 'Programming I'. Moreover, the graph shows a number of less

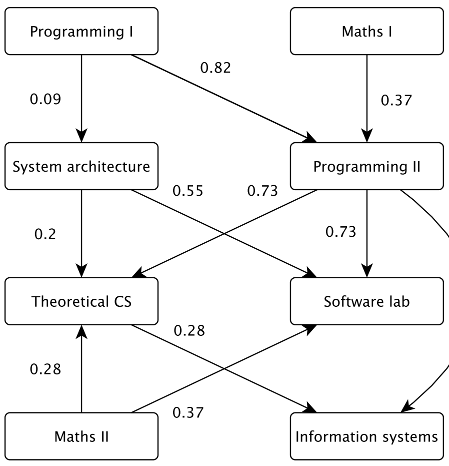


Figure 1: Excerpt of a course dependency graph, based on *Mann-Whitney U-test* with significance level of 0.0001, representing the dependencies between most of the basic courses in CS curriculum at Saarland University.

obvious relations between courses (e.g. 'Programming II' and 'Theoretical Computer Science').

4.2 Grade prediction

We use a *collaborative filtering* [10] approach to predict student performance. One advantages of this approach is that no imputation of missing entries is necessary but the optimization only runs over existing entries.

We associate with each student i and course j an n-dimensional feature vector, s_i and c_j, respectively. The predicted performance is the cross-product of both vectors, i.e.

$$f(i,j) = \langle s_i, c_j \rangle = \sum_{k=1}^{n} s_{i,k} c_{j,k} \,,$$

which we call the *predictor* function. Let $g_{i,j}$ be the performance of a student i in course j and let $\mathcal{G}_t$ denote the set of all known performances of students up to semester t. Then the standard loss is the regularized MSE, i.e.

$$L(S,C,t) = \sum_{g_{i,j} \in \mathcal{G}_{t-1}} \left(f(i,j) - g_{i,j}\right)^2 + \lambda h(S,C)$$

with regularization term

$$h(S,C) = \sum_{i \in S} \|s_i\| + \sum_{j \in C} \|c_j\| \,,$$

where S is the set of all students and C the set of all courses.

4.2.1 Contextual Information

The above loss metric only depends on information about the students' performances, i.e. their grades. However, the *context* of a performance can contain vital information. Usually, in the context of student records a wealth of data is readily available. This includes meta-data of a student such as age, gender, and nationality and data regarding the progression of the student throughout study programs. Moreover, information regarding the course, such as the lecturer, is typically known.

A standard and straight-forward, approach to include such information is to *pre-filter* data [10]. This entails partitioning data along contextual criteria and then training a model for each subset. Here, the only performed pre-filtering is to take only computer science Bachelor students into account. Other partitionings, e.g. partitioning along the semester, have not improved predictive quality.

Further contextual information is included explicitly in the model as follows. The predictor f is augmented by linear terms for contextual features. Categorical features, such as teachers, are one-hot encoded. Continuous features are centered to zero mean and unit variance. In principle we can introduce these additional linear parameters for both, courses and students, but it turns out that the best results are achieved if we associate features with courses. Given the large number of contextual features it proved advantageous to set up a feature selection pipeline in which certain features are identified for each course. Specifically, features were identified by using a 5-fold cross-validated *recursive feature elimination.* Therein features are iteratively removed according to their coefficient in a linear model. The cross-validation is used to determine the number of features kept. Thus, the predictor becomes

$$\tilde{f}(i,j,t) = \langle s_i, c_j \rangle + \langle ctx(i,j,t), c_j^{ctx} \rangle \,,$$

where *ctx* is the performance context according to the above feature selection pipeline. Consequently, the parameter vector for course j becomes

$$\tilde{c}_j = (c_{j,1}, c_{j,2}, \ldots, c_{j,n}, c_{j,1}^{ctx}, \ldots, c_{j,m_j}^{ctx})$$

and m_j is the number of features selected for the context of a performance in course j.

Another key property to be considered when working with past performances is the temporal distance to the current time. A performance achieved one semester ago should be considered more important than one five semesters ago [9]. Therefore it is natural to add a temporal decay to the loss function. Considering the semester t' of a specific performance $g_{i,j,t'}$, we can multiply an exponential decay function. Thus, the now time-dependent loss is

$$L(S,C,t) = \sum_{g_{i,j,t'} \in \mathcal{G}_{t-1}} e^{-\alpha \cdot (t-t')} \left(\tilde{f}(i,j,t) - g_{i,j,t'}\right)^2 + \lambda h(S,C) \,, \quad (1)$$

where $\alpha > 0$ is the temporal decay parameter.

4.2.2 Minimization

The non-linear minimization problem in Eq. (1) is of high dimensionality because of the parameter vectors s_i and c_j for $i \in S, j \in C$. It can most effectively be achieved using stochastic gradient descent techniques with adaptive learning rates, because for this approach course vectors stabilize more quickly. Specifically, we used the Adagrad algorithm [4], which avoids strong alteration of frequently considered parameters, which is the case for many course parameters, while seldomly encountered parameters may be altered more, which is fitting for student parameters. We fixed a batch size of 1000 and performed 500,000 iterations of the algorithm. Each minimization is performed for 5 different initial random values. The value according to the smallest training loss is selected. This was performed for all semesters in a grid search over different dimensionality parameters and regularization parameters, i.e. for parameter tuples (λ, n). Before minimization the data was normalized along the lectures to zero mean and unit variance.

4.2.3 *Evaluation*

The most natural approach to evaluate the model is to split the data by semesters. Given a fixed semester t the data up to (including) semester $t-1$, i.e. $\mathcal{G}_{t-1}$, is used as a training set. The data of semester t, i.e. $\mathcal{G}_t \setminus \mathcal{G}_{t-1}$ is used as a test set.
The measures of quality we use are the mean absolute error (MAE) and the root mean square error (RMSE). As a baseline we provide the RMSE and MAE for the mean predictor with respect to both, the students and the courses in Table 1.
In the evaluation of the context-free model, we see, that low-dimensional models (i.e. models with only few features) perform best. The absolute values of these errors are further improved by pre-filtering the data considered. If, for example, only Bachelor computer science students are considered the test error decreases. The decay factor leads to an improvement. For example, for $n = 1$ and $\lambda = 0.1$ the MAE decreases from 0.856 to 0.852. In Figure 2 the prediction results for the importance decay $\alpha = 0.1$ are shown. Given this loss function, the one-dimensional, less regularized model outperforms the others in terms of both, the MAE and the RMSE. The inclusion of contextural information leads to a further reduction, such that for $n = 1$ and $\lambda = 0.1$ the MAE is 0.8459, while the RMSE is 1.0904.

Table 1: The RMSE and MAE for the mean predictors along the student and the course axis, respectively.

	MAE	RMSE
course	1.1130	1.3311
student	0.9268	1.1883

5. RECOMMENDATION SYNTHESIS

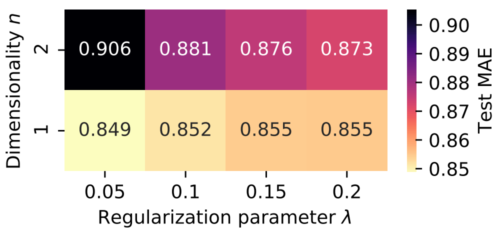
The recommendation combines the course dependency graph, the grade prediction, and constraints based on the study regulation in order to compute a *recommendation score*. A larger score corresponds to a stronger recommendation.

5.1 Combining the Components

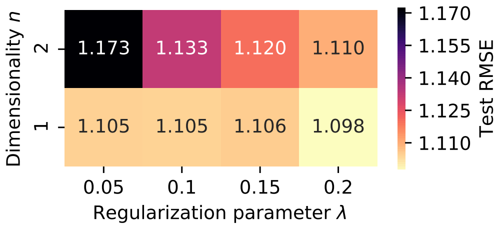
The recommendation score for a course j w.r.t. a student i combines several criteria, namely the preparedness for j, the general merit of j, and the predicted performance of i in course j.

Let R_i denote the set of courses that student i has finished within the last t semesters. Now, for each course $j \in C \setminus R_i$, we sum over the weights of the edges of the course dependency graph that start in some course $j' \in R_i$ and end in j. This value is an approximation for the preparedness $p_{i,j} \in \mathbb{R}_{\geq 0}$ of the student w.r.t. course j.

For the general merit of a course, we use the out-degree of the course $\deg^+(j)$ in the graph as an approximation of its benefit towards other courses. Note that this criteria is especially relevant for first-year students as for them nodes with



(a)



(b)

Figure 2: The MAE (a) and RMSE (b) for different dimensionalities n and regularization parameters λ. The models were trained and tested on Bachelor CS students only. The loss is weighted by time with $\alpha = 0.1$.

higher out-degree provide a good starting point. Further note that for such students, $R_i = \emptyset$ and the grade prediction can only give average values as no information about their previous performance is available.

To incorporate information about the predicted performance, we transform the predicted grades $\hat{g}_{i,j}$, such that good grades map to large values and poor grades to small values, i.e., we consider the value $(5 - \hat{g}_{i,j})/4 \in [0, 1]$.

We parameterize these factors into a linear model, that gives us a raw, unfiltered recommendation value

$$r'_{i,j} = c_p p_{i,j} + c_g(5 - \hat{g}_{i,j})/4 + c_m \deg^+(j), \qquad (2)$$

where $c_p, c_g, c_m \in [0, 1]$ provide a weighting for the three factors, i.e., $c_p + c_g + c_m = 1$.

We finally filter the recommendations as follows. The choice of courses is constrained by study regulations. Thus, for a given student i, some courses may not contribute towards completion of the program or she may not be able to enroll in them ('not allowed'). Thus, the final recommendation value is a product of the raw value $r'_{i,j}$ and a function value $reg(i, j)$, where

$$reg(i,j) = \begin{cases} 1 & j \text{ is part of program} \\ 0 & j \text{ not allowed} \\ c_e(i) & \text{otherwise} \end{cases}$$

This introduces a further parameter $c_e(i) \in [0, 1]$ associated with courses that are not necessary to achieve the degree but may lead to an improvement of the final grade or may be interesting to the student. E.g. a student of bioinformatics may choose $c_e(i) = 0.5$ to get also recommendations for computer science courses that are not part of the bioinformatics

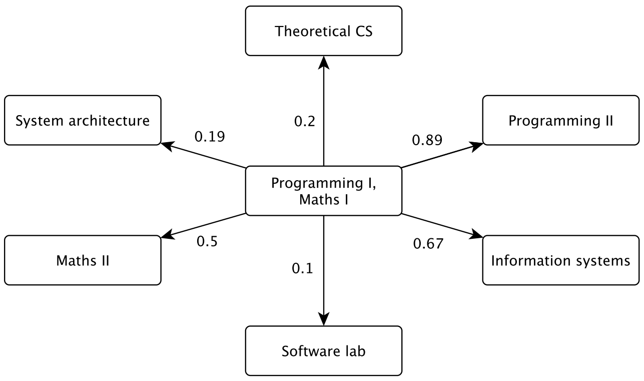


Figure 3: Example of a recommendation graph, based on the dependency graph given in 1. ‘Programming I’ and ‘Maths I’ have been passed already and the edge weights have been updated accordingly. The recommendation values were computed with $c_p = 0.76$, $c_g = 0.21$ and $c_m = 0.03$.

program. However, the default value is $c_e(i) = 0$.

Hence, the overall recommendation value of course j is

$$r_{i,j} = \big(c_p p_{i,j} + c_g(5 - \hat{g}_{i,j})/4 + c_m \deg^+(j)\big)\, reg(i,j) \qquad (3)$$

with weight parameters by c_p, c_g, c_m.

To illustrate the influence of the different factors, we consider the following example. Suppose a first-year student in the winter semester uses the system to compose his first curriculum. We do not have any performance knowledge about the student, so this is a cold start scenario. Reconsider the dependency graph in Figure 1. Because of the high out-degrees, we recommend ‘Programming I’, ‘Maths I’ and ‘Theoretical CS’. The student successfully attends the first two of these courses in the following winter semester. Now we are able to incorporate the achieved grades in our prediction model. The now computed recommendation values per course are visualized as star graph shown in Figure 3. Finally a valid suggestion for the next semester based on the recommendation values is a combination of ‘Programming II’, ‘Information systems’ and ‘Maths II’. In general, at the beginning of every semester, we can provide the student with a personalized curriculum by compiling a list of lectures based on their recommendation score.

5.2 Evaluation

We now assess how similar our recommendation are to the actually selected courses of the students. Again, we separate the student data by semesters, such that recommendations are only based on data of previous semesters. To define the metric, let $\mathcal{T}$ be the set of semesters, $\mathcal{S}_t$ the set of students who took some course in semester $t \in \mathcal{T}$. Further, given some semester t, let $C_{sel}^{i,t}$ be the set of courses in which student i was enrolled and let $C_{rec}^{i,t}$ be the set of recommended courses for student i. We adopt a top-k recommendation policy in which we recommend only the k courses with the highest recommendation value. Moreover, we only take into account lectures which were available in the given semester and study program.

To approximate the conformity of our recommendations we consider the *conformity score*

$$1 - \frac{1}{|\mathcal{T}| + |\mathcal{S}_t|} \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{S}_t} \frac{\min(k, |C_{sel}^{i,t}|) - |(C_{rec}^{i,t} \cap C_{sel}^{i,t})|}{\min(k, |C_{sel}^{i,t}|)},$$

where the second term calculates the average ratio of the number of courses that have been selected by the student but were not recommended or that were recommended but not selected. So we end up with a score, indicating the congruency of our recommendations with the student’s actual course selections.

We evaluated the conformity score w.r.t. several combinations of the recommendation parameter values of Eq. (3). The considered recommendation sizes are 4 and 6 courses, since for most students this is a realistic balance between study progression and manageable a workload.Since we are interested in the relationship between the conformity score and the distribution of the parameters, in the first place we either fix c_p or c_g to 1 while the rest stays at zero which captures the performance of a single component of our approach. Moreover, we look for the best combination of both, course dependency graph (c_p) and grade prediction (c_g). The third parameter $c_m = 1 - c_p - c_g$ results from the choice of the first two, which makes the search two-dimensional.

Our results in Table 2 show that with increasing k the conformity grows as more courses are recommended. The first two columns of the table point out that the course dependency graph has a higher explanatory value for the recommendation than the grade prediction. A recommendation only based on the performance hardly achieves a value exceeding 50 percent while course dependency alone reaches 70 percent. Therefore it is clear that c_p has to be determined significantly larger than c_g. This observation is approved within the third column as in all top-k recommendations we reached the best conformity with $c_p \approx 0.76$, $c_g \approx 0.21$ and $c_m \approx 0.03$.

According to these scores our recommendations and the choices of the students have an average overlap of about 70 percent. Hence, there are recommended courses that the student did not choose. An example for this case is given by the core lecture ‘Embedded Systems’. We recommended this course to 89 students, while only 4 of them actually took the course in the corresponding semester. As opposed to mathematically demanding lectures such as ‘Complexity Theory’, which is only recommended for a small set of very strong students, this course seems to be a good choice for many students but is taken only by few. Moreover, in one semester the number of recommendations for basic courses was about 200 while only 90 students actually attended the courses. This could be related to the fact that many students withdraw from courses after a few weeks when they feel that the course is too demanding for them. In this case, the data does not show their trial for this course.

6. CONCLUSION

We proposed an approach that gives personalized course recommendations for students in order to improve the obtained grades and to decrease the time-to-degree. We combined a course dependency graph and performance predictions to

Table 2: The conformity score for different valuations of the recommendation value parameters (c_p, c_g, c_m) and different top-k recommendation policies.

top-k	$(c_p, c_g) = (1, 0)$	$(c_p, c_g) = (0, 1)$	$(c_p, c_g)^*$
4	0.5913	0.3857	0.6349
5	0.6580	0.4564	0.6962
6	0.7138	0.5326	0.7432

determine a recommendation value for each course. We assumed that only the top-k courses are given as a personalized curriculum for a student and tested their conformity to the actually selected courses of the student. This, however, does not indicate that our approach significantly improves the students' grades or time-to-degree as we expect that students do not make optimal choices.

An interesting insight from our results is that the course dependency graph seems better suited for course recommendation than grade prediction even though it is only based on aggregated information and does not consider any meta data. From this result it seems that students tend to focus more on a course ordering that older students established rather then selecting according to their own confidence or skill.Another interesting result is the large overlap (around 70 percent) of recommended and chosen courses. Moreover, some courses are not taken by students even though our model indicates that they would lead to an improvement in performance.

The model itself is flexible in the sense that one can easily adjust or extend it by changing the recommendation formula and/or incorporate more information to make the grade prediction more precise. A possible extension is the integration of more personalized information given by the student before calculating their recommendations. For example a student is more interested in practical lectures, so she uses an interface to let the system know. Thus, we would be able to give courses of this category a positive effect on their recommendation value. The challenge here is to separate the courses into appropriate categories, since the way a course is designed strongly depends on the lecturer and other factors.

To evaluate the system, it would be interesting to monitor a sufficiently large number of students during their studies that choose only recommended courses or at least is exposed to the course recommendations. An easier evaluation would be possible with a simulation of hypothetical student traces according to our grade prediction approach, where in each semester we assume that a student chooses only recommended courses.

7. REFERENCES

[1] R Asif, A Merceron, S Abbas Ali, and N Ghani Haider. Analyzing undergraduate students' performance using educational data mining. *Computers & Education*, 113:177 – 194, 2017.

[2] M Baron. *Probability and Statistics for Computer Scientists*. Chapman & Hall, 2014.

[3] CM Chen, CY Liu, and MH Chang. Personalized curriculum sequencing utilizing modified item response theory for web-based instruction. *Expert Systems with applications*, 30(2):378–396, 2006.

[4] J Duchi, E Hazan, and Y Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[5] C Hansen, C Hansen, N Hjuler, St Alstrup, and C Lioma. Sequence modelling for analysing student interaction with educational systems. In *Conference on Educational Data Mining*, pages 232–237, 2017.

[6] A Karatzoglou, X Amatriain, L Baltrunas, and N Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Conference on Recommender systems*, pages 79–86. ACM, 2010.

[7] A Polyzou and G Karypis. Grade prediction with models specific to students and courses. *International Journal of Data Science and Analytics*, 2(3-4):159–171, 2016.

[8] S Ray and A Sharma. A collaborative filtering based approach for recommending elective courses. In *International Conference on Information Intelligence, Systems, Technology & Management*, pages 330–339. Springer, 2011.

[9] Z Ren, X Ning, and H Rangwala. Grade prediction with temporal course-wise influence. *Conference on Educational Data Mining*, 2017.

[10] F Ricci, L Rokach, B Shapira, and PB Kantor. *Recommender systems handbook*. Springer, 2015.

[11] M Saarela and T Kärkkäinen. Analysing student performance using sparse data of core bachelor courses. *Journal of educational data mining*, 7(1), 2015.

[12] A Slim, GL Heileman, J Kozlick, and CT Abdallah. Employing markov networks on curriculum graphs to predict student performance. In *Machine Learning and Applications, Conference on*, pages 415–418. IEEE, 2014.

[13] SE Sorour, T Mine, K Goda, and S Hirokawa. A predictive model to evaluate student performance. *Journal of Information Processing*, 23(2):192–201, 2015.

[14] M Sweeney, J Lester, and H Rangwala. Next-term student grade prediction. In *Big data*, pages 970–975. IEEE, 2015.

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

[16] A Töscher and M Jahrer. Collaborative filtering applied to educational data mining. *KDD cup*, 2010.

[17] H Vossensteyn, A Kottmann, B Jongbloed, F Kaiser, L Cremonini, B Stensaker, E Hovdhaugen, and S Wollscheid. Dropout and completion in higher education in europe: Main report. 2015.

[18] J Zimmermann, KH Brodersen, HR Heinimann, and JM Buhmann. A model-based approach to predicting graduate-level performance using indicators of undergraduate-level performance. *Journal of Educational Data Mining*, 7(3):151–176, 2015.