Investigating Performance in a Blended SPOC

Wan Han[1,2], Ding Jun[1], Liu Kangxu[1], Gao Xiaopeng[1,2]
School of Computer Science and Engineering[1], Honors College of Beihang University[2]
Beihang University
Beijing, China
{wanhan, dingjun, liukangxu, gxp}@buaa.edu.cn

***Abstract*—In this paper, we describe how to investigate performance in a blended SPOC (small private online course). For the quantitative research, we build predictive models of students' performance each week in a SPOC. We document a performance prediction methodology from raw logging data extracted from Open edX platform to model analysis. These logging data were collected from Computer Structure Lab Course offering at Beihang University. We show how to extract meaningful information from the learning related educational data we gathered. 28 predictive features extracted for 377 students, and our model achieved an AUC (area under curve) in the range of 0.62-0.83 when predicting one week in advance. An early warning system is established to identify at-risk students in the SPOC, especially for the blended lab course. Furthermore, we could use the most important features to form the assessment for each student during the semester.**

Keywords—SPOC; student performance prediction; study behavior analysis; educational data mining; at-risk students; assessment and evaluation; early warning system

I. INTRODUCTION

EdX has designed and built an open-source online learning platform (Open edX) for online education. In addition to offering online courses, participating universities are also committed to researching how students learn and how technology can transform learning both on-campus and online throughout the world.

The biggest success for MOOCs (Massive Open Online Courses) and their greatest problem are the same - the huge numbers attracted. Most of the signs-up will drop-out without completing courses. So how to use what we're learning in MOOCs, and the technology we're creating with MOOCs, to improve education on campus?

MIT and Harvard prioritized the development of an introductory computer science course, which was first offered as a MOOC in the fall of 2012 and then adapted as a blended learning course at the two community colleges [1]. The course, based on 6.00x: Introduction to Computer Science and Programming, provides a broad introduction to computational thinking targeted at non-computer science majors and introduces students to the foundations of the Python programming language. In the blended (or flipped) learning environment, the in-class lecture principally occurs online through video and other interactive media and homework is moved to the class meeting time. This SPOC maximized the opportunity for contextualization and application of the lessons directly through practice and problem solving with peers and instructor.

HarvardX offered two SPOCs in Fall 2013 - a Law School course titled HLS1x: "Copyright," chose from 4,100 applicants worldwide to form the 500-student online class. Another new HarvardX courses include HKS211.1x, "Central Challenges of American National Security, Strategy, and the Press: An Introduction," a Kennedy School module SPOC [2] that had limited enrollment for certain features, such as discussion boards.

A SPOC created edX-based materials to support on-campus course. Even if the goal is to create a MOOC, "debugging" the materials by deploying as a SPOC first is a great idea. As Armando Fox mentioned in [3], in a "virtuous cycle," pedagogies can be beta-tested in a campus setting, deployed to MOOC learners once refined, and the data collected and analyzed from those learners are used to improve the material, benefiting the classroom learners.

Our basic idea is to use MOOC-style video lectures and other online features as course materials in an actual on-campus course. By assigning the lectures as homework, the instructors are free to spend the actual class period asking and answering students' questions, gauging what they have and haven't absorbed. We developed 'virtual lab' technology for lab-based courses [4], such as Digital System Design and Computer Structure. In Fall 2016, this approach combines two software simulators (MARS and Logisim) for the lab with algorithms that automatically grade student solutions and provide feedback to students.

The edX LMS and Studio are instrumented to enable tracking of metrics and events of interest [5]. These data can be used for educational research, decision support, and operational monitoring.

The most notable feature of MOOCs is it's easily to be accessed but high dropout rate, and all students' activity are recorded in MOOC's server logs. Many researchers have studied extracting students' learning features from the log, then analyzed the data using data mining and machine learning methods. They also built models to predict students' activity or performance in the future.

978-1-5386-0900-2/17/$31.00 ©2017 IEEE 12-14 December 2017, The Education University of Hong Kong
2017 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)

Based on stochastic process and combined with the EM algorithm, Wang, F. [7] presented a nonlinear state space model for identifying at-risk students. Some researchers noticed that the predicting methods for active students and absent should be different, and they designed two predictors for different mode [10]. Various machine learning models were applied in predicting dropout in MOOCs. Logistic regression algorithm was explored for predicting failure or dropout weekly [11] [18]. Principal component analysis (PCA) and support vector machine (SVM) are also effective in solving this problem [13]. In [17], the predicting model is based on RNN. Srilekshmi, M. [12] demonstrated the application of association rule learning in identifying students at-risk.

On the other hand, some researches focus on how to predict students' performance by using study-related data. Stapel, M. [8] presented an ensemble method to predict students' performance, which includes six classification algorithms. Elbadrawy, A. [9] developed multi-regression models based on regression algorithms for predicting, and Ren, Z. [14] designed different kinds of features based on MOOC courses' characters, which improved the performance of their predictor. Multiple linear regression [15] and Naive Bayes Classifier [16] were also used to predict students' performance. In addition to study-related data, social behavior data is helpful in predicting [6].

In this paper, we describe the performance prediction problem, and present models we built. A summary of which features played a role in gaining accurate predictions is presented. The most fundamental contribution is the design, development and demonstration of a performance prediction methodology, from raw logging data to model analysis, including data preprocessing, feature engineering, model evaluation and outcome analysis. We will next generalize this methodology to find students at-risk each week in Fall 2017 at Beihang University.

II. PREDICTION PROBLEM DEFINITION

Computer Structure Lab Course offering at Beihang University in Fall 2016 was composed of 3 tutorials and 9 projects, the curriculum is shown in TABLE I. . Learners studied the tutorials from week 1 to week 6, and instructors released project 0 at week 7. As we described in [4], we found it was important for learners to move on only after they'd mastered the core concept. Students started one project and they need to pass the test in class. When they mastered corresponding content related to one project, and then they could be awarded to the next project.

For instance, at the beginning of week 10, learners who passed test for Project 2, first start the single cycle CPU design using Logisim, which need support MIPS-Lite1 including 7 instructions. At the end of week 10, there was a test for Project 2. In the test, one instruction that does not include in MIPS-Lite 1, must add into students' basic design. This new design should pass through the auto-testing system. Then there was a face-to-face discussion with teacher or TA. In this discussion stage, several questions related the project are discussed between the learner and instructor. Once the instructor is satisfied that the learner has mastered all elements of this project level, then the learner could move forward to the next project.

TABLE I. COURSE CURRICULUM IN COMPUTER STRCTURE

	Project Name	Project Content	Release week
P0	The basic Verilog-HDL/ Logisim practice	Master the language grammars, and could test the function/ timing of the project using testbench. Design ALU and Comparator.	week7
P1	CPU component design using Verilog-HDL/ Logisim	Design finite state machine.	week8
P2	MIPS assembly language exercises using Mars	Learn how to code and debug assembly programs in the MARS	week9
P3	Single Cycle CPU design using Logisim	This single cycle CPU should support the MIPS-Lite1, 7 instructions. Learn how to use sub-circuits to demonstrate the schematic, datapath and control	week10
P4	Single Cycle CPU design using Verilog-HDL	Support MIPS-Lite2 instructions: {MIPS-Lite1, 3 more instructions}	week11
P5	Pipelined processor design with full hazard handling	Separate the hazard unit from controller module Support MIPS-Lite2 instructions	week12
P6	Pipelined processor design support MIPS-C3 instructions	MIPS-C3 instructions: {MIPS-Lite2, 40 more instructions}	week14
P7	MIPS micro-system-V1 support exceptions and interrupts	Micro-system including MIPS-processor, bridge and timer. Support MIPS-C4 instructions: {MIPS-C3, ERET, MFC0, MTC0}	week15
P8	MIPS micro-system-V2 support exceptions and interrupts	Micro-system including micro-system-V1, 8bit-LED segment, 32bit-toggle switch and RS232-communicate-system. Support MIPS-C4 instructions	week16

A. Performance Definition

As we elaborated above, learners would be in different project's test at the end of each week in class. For example, at the end of week 7, there was a test for Project 0. If learners passed the test would enter P1 at the beginning of week 8, and then participate the test of P1 at the end of week 8. For the others who failed the P0 test at the end of week 7, need to redo P0 test at the end of week 8.

Here our performance prediction is to predict whether the learner could pass their test at the end of each week according to their study behavior.

B. Temporal Prediction

In our SPOC, course content was assigned on a weekly basis, where each week corresponded to an Open edX chapter (module). Here we define time slices as weekly units. Time slices started the first week in which in class test was offered (week 7), and ended in the 16th week, after the final test had closed. So we could use the logging data from week 1 to week 6 to predict the learners' performance at week 7.

Furthermore, we used lead represents how many weeks in advance to predict performance. We assign the performance label (x1, 0 for unpassed the test or 1 for passed the test) of the lead week as the predictive problem label. Lag means use how many weeks of historical variables to classify.

We use a realistic scenario to explain the predictive model's application. The instructor could use the data from week 1 to the current week i to make predictions. The model will predict existing learner performance during week (i + 1) to week 16. As shown in Fig. 1, current week is week 7, and we use the logging data from week 1 to week 7 to predict the learners' performance at week 12 with lead equals to 4 and lag equals to 7.

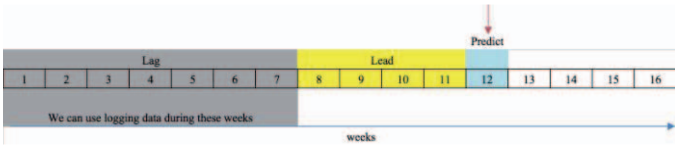

Fig. 1. Learners' weeks data used in a lead 4, lag 7 prediction problem

III. Feature Engineering

We extracted features on a per-learner basis from logging data. In Open edX system, events are emitted by the server and the browser to capture information about interactions with a course and are stored in JSON documents.

A. *Events Observed in the Logging Data*

Learners' activity events including their interactions with any resources in the learning management system. For instance, when a user use a browser to stream video files, the browser emits the events. The video interaction events include ‘load_video’, ‘pause_video’, ‘play_video’, ‘show_transcript’, ‘speed_change_video’, and ‘stop_video’, which are according to the user interacts with a video.

In the logging data, it also recorded ‘who’ interact with the resources at ‘what time’. If the learner attempt to interact with a problem, there are several events will be emitted. These events include ‘problem_check’, ‘problem_rescore’, ‘problem_save’, ‘showanswer’ and so on.

There are many other resources such as discussion forum, etext and so on. Learners' access to these resources would trigger corresponding events and then would be written in the log.

B. *Feature Definition*

In our paper, we did not use the non-behavioral attribute such as a leaner’s age, gender and others. Instead, we used some features that would show different style of learning habits. One type of behavioral variables is based on the learner’s interaction with the educational resources, including time spent on resources and problem / homework. These features are based on observed event as shown follows:

- x1- week performance, whether the student has passed week test or not.
- x2- total_duration, total time spent on all resources.
- x3- number_forum_posts, number of forum posts.
- x4- total_lecture_duration, total time spent on lecture resources.
- x5- average_length_forum_post, average length of forum posts.
- x6- number_distinct_problems_submitted, number of distinct problems attempted.
- x7- number_submissions, number of submissions.
- x8- number_distinct_problems_submitted_correct, number of distinct correct problems.

As Colin Taylor described in [18], taking the extra effort to extract complex predictive features that require relative comparison or temporal trends, rather than using the direct covariates of behavior, is one important contributor to successful prediction. For instance, we create an average number of submissions per problem for each learner (x9). Then we compare a learner’s x9 value to the distribution for that week. Feature x16 is the percentile over the distribution and x17 is the percent as compared to the max of the distribution. These complex behavioral features and their brief descriptions are listed as follows:

- x9- average_number_submissions, (x7 / x6).
- x10- observed_event_duration_per_correct_problem, (x2 / x8).
- x11- submissions_per_correct_problem, (x6 / x8).
- x12- average_time_to_solve_problem, (average(max(submission.timestamp) - min(submission.timestamp) for each problem in a week)).
- x13- observed_event_variance, variance of a student's observed event timestamps.
- x14- max_observed_event_duration, duration of longest observed event.
- x15- number_forum_responses, number of forum responses.
- x16- average_number_of_submissions_percentile, a student's average number of submissions / other students.
- x17- average_number_of_submissions_percent, a student's number of submissions / maximum average number of submissions.
- x18- number_finished_problem_submissions, number of correct problem that had been submitted.
- x19- correct_submissions_percent, (x8 / x7).
- x20- average_predeadline_submission_time, average time between a problem submission and problem due date over each submission.

Furthermore, study habits related behavioral features were extracted. For instance, feature to describe whether learners begin doing the problem / homework soon after it was released, and features to characterize the learners that submit problem / homework in timely fashion or at last minute fashion. We also used the academic performance related which prior to the course – freshmen year GPA. These behavioral features including:

- x21- time_till_first_check, sum of all problem the time between Problem_first_check and Problem_first_get.
- x22- time_on_problem_atomic, sum of all time intervals dedicated to the problem.
- x23- time_on_problem_molecularis, time between first problem_get to last problem_check.

- x24- problem_finish_time_pre_start24h, sum(problem start time + 24h - problem finished time), if the learner finish all the problem correctly in the first 24h after the project issued.
- x25- problem_finish_time_pre_start48h, sum(problem start time + 48h - problem finished time), if the learner finish all the problem correctly in the first 48h after the project issued.
- x26- problem_finish_time_pre_deadline24h, sum(problem due time - problem finished time), if the learner finish all the problem correctly in the last 24h before deadline.
- x27- problem_finish_time_pre_deadline48h, sum(problem due time - problem finished time), if the learner finish all the problem correctly in the last 48h before deadline.
- x28- total_chapter_material_time_before_submit, chapter_video/etext access time before the problem submit.
- x29- time_first_attempt, min(time_first_problem_get, time_first_html etext_access) - project_issue_time.

C. Data Flattening

To build predictive models, we utilize a common approach of flattening the data- assembling the features from different weeks as separate variables.

As shown in Fig. 2, we assembled the features from week 1 with week 2 to predict the performance at the end of week 15. This prediction problem corresponds to a lead of 12 and a lag of 2.

IV. Predicting Week Performance

We first used logistic regression as our binary predictive model. It calculates a weighted average of a set of variables as an input to the logit function. There are different coefficients for the feature values. For the binary classification problem, the output of the logit function becomes the estimated probability of a positive training example.

When applying the logistic regression to learner week performance prediction. We used the 28 features we described in section 3 to form the feature vectors, and maintained the week performance value as the label.

A. Learning

The objective of training in this model is to find a set of coefficients well suited to the data. The training process involves passing a set of covariates and a corresponding binary label associated with the covariates.

First, a random set of coefficients are chosen. During each iteration, an algorithm is used to find the gradient between what the coefficients predict and what they should predict, and then updates the weights. This process repeats until the change in the coefficients is sufficiently small, and they represent the final trained model.

After training, the output of the logit function should predict higher probabilities for the positive class examples and a lower probability for the negative class examples.

B. Predicting Performance

When evaluating the classifier's performance. A testing set comprised of untrained covariates and labels evaluates the performance of the model as following steps:

The logistic function learned is applied to each data point and the estimated probability of a positive label is produced. And then a decision rule is applied to determine the class label for each probability estimate. Given the estimated labels for each data point and the true labels we calculate the confusion matrix, true positives and false positives and then obtain an operating point on the ROC curve. Then evaluate the area under the curve and report it as the performance of the model on the test data.

As we mentioned in II.B, we need to present the results for multiple prediction problems for different week simultaneously. Here means for each week during our course, we want to predict the students' week performance using different historical data. The heat map of a lower right triangular matrix is assembled as shown in Fig. 3.

The x-axis of Fig. 3 is the week for which predictions are made in the experiment, while y-axis is the number of the how many week data we use for the prediction (lag). The color shown the area under the curve for the ROC the current model achieved.

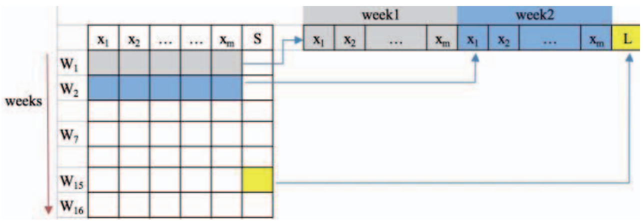

Fig. 2. Data flattening process

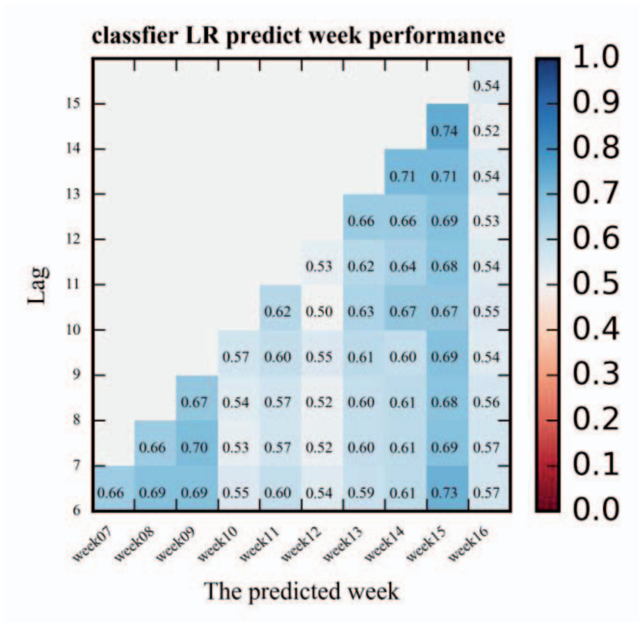

Fig. 3. Logistic regression predict week performance.

We employed cross validation in all of our predictive modeling. Some partitions are used to construct a model, and others are used to evaluate the performance. In our work, we used K-fold cross validation. We randomly divides the dataset into K partitions, and each model is constructed using K-1 of partitions, and the model is evaluated using the last unused partition. Considering there is only 377 samples in our data set, we employed 3-fold cross validation and use the average of the ROC AUC over the folds as evaluation metric.

C. Feature Importance

We utilized randomized logistic regression methodology to identify the relative weighting of each features. The model training process follows the steps below:

Step 1: Sample without replacement 75% of the training data.

Step 2: Training a logistic regression model on the sub-sampled data.

Step 3: For every feature evaluate $b_{is}= \mu(w_i ,th)$ where μ is a unit step function and w_i is the coefficients for covariate i and th is the threshold we set to deem the feature important. This is set at 0.25.

Repeat steps 1-3 for 200 times.

Estimate the importance of the covariate i by $\sum_s b_s^i$.

The feature F_x importance in one lag- lead combination is

$$y = \frac{\sum_{j=1}^{lag} F_{xj}}{\sum_{i=1}^{28}\sum_{j=1}^{lag} F_{ij}}$$

F_{ij} represents the future i's importance (in week j) produced from randomized logistic regression model.

For each feature, we sum of all its importance in 55 lag-lead combination in order to produce the week-invariant importance.

As shown in Fig. 4, top features that had the most predictive power include whether learners interact with the resources more time(max_observed_event_duration), learners' interaction with the problems (average_number_of_submissions_percentile), study habits (time_first_attempt, problem_finish_time_pre_start24h, problem_finish_time_pre_start48h).

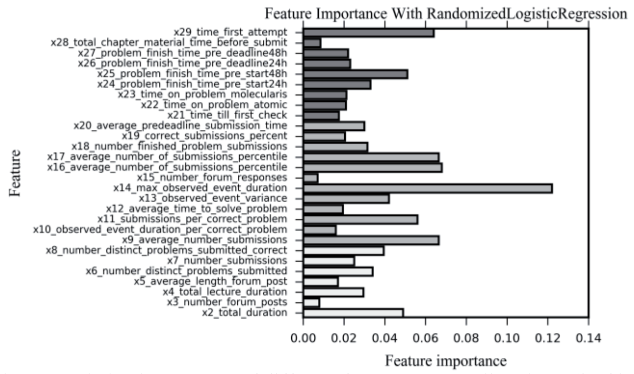

Fig. 4. Relative importance of different features across all variants (lag/ lead)

V. USING SVM TO BUILD AN EARLY WARNING SYSTEM

Fig. 5 summarized the AUC of the receiver operating characteristic over each lead and lag combination when using SVM to predict the learners' week performance.

Overall, the average SVM predicted week performance achieved accuracy at ~ 0.70. Some experiments, such as a lag of 9, predicting week9 achieved accuracy as high as 0.82. With a lead of one, this model resulted in an AUC between 0.61 to 0.82. Thus, we can surmise that the extracted features are capable of predicting performance in each week, especially when the prediction week is near the lag week.

We added freshmen year GPA as their academic performance related which prior to the course – this course is a second-year course for undergraduates. As shown in Fig. 6, the average of AUC achieves ~0.01 higher.

If these features feed into model predictions hold true even for other courses, this prediction could be used to measure whether the learner is at-risk at the predictive week. And we could offer reminder email with some recommend resources to the learners who are at-risk in our early warning system.

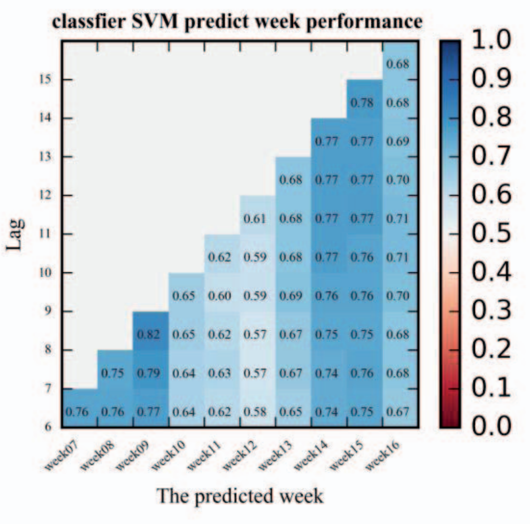

Fig. 5. Support vector machine predicting results

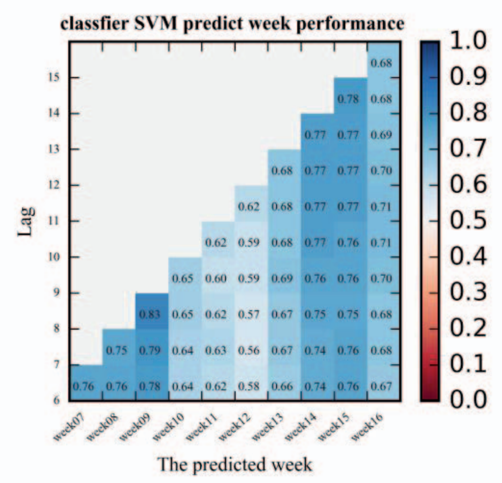

Fig. 6. Predicting results when adding freshman year GPA

VI. Student Assessment

We form a five-dimension assessment to evaluate the performance of each student in 2016 Fall semester. The five dimensions are compound participating features which had been found important: Video watching, Text reading, Participation, Problem Solving and Discussion.

As shown in Fig. 7 and Fig. 8, student who passed project 7 in the course have much higher score distribution than the student who were failed in the course in all assessment dimensions. Fig. 9 shows the score distribution of the student who passed project 6 in the five-dimension assessment. One thing need to be mentioned is that our blended SPOC had small amount videos, and e-texts almost covered all the content of videos.

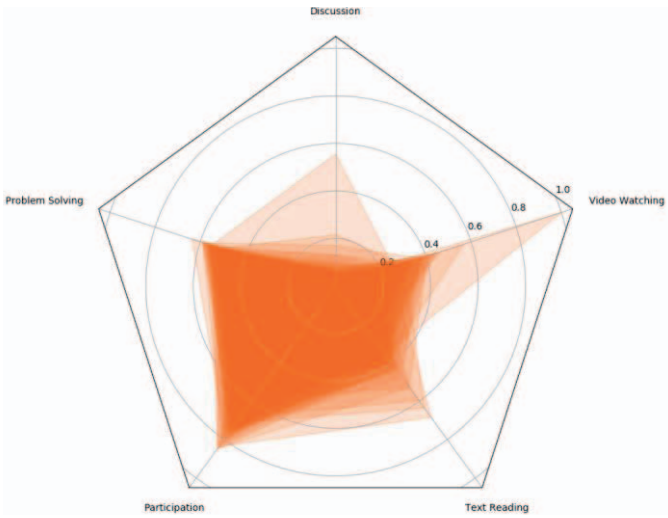

Fig. 7. The score distribution of student who failed in project 5

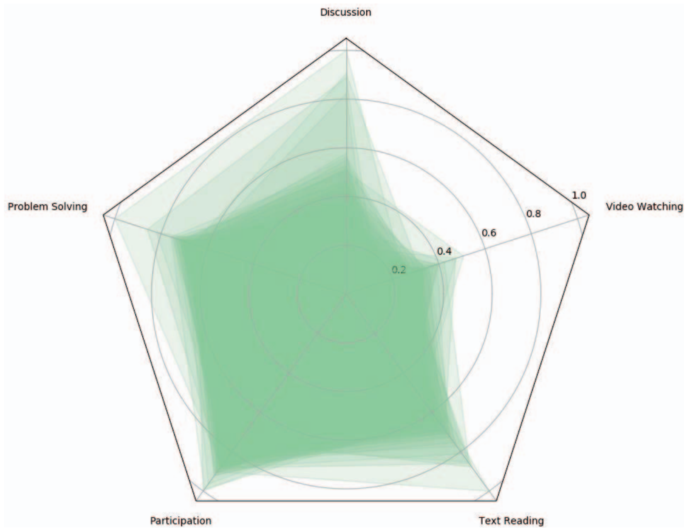

Fig. 8. The score distribution of student who passed project 7

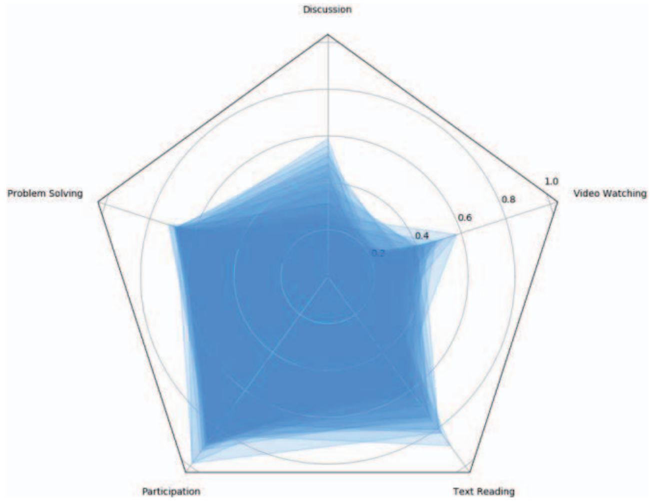

Fig. 9. The score distribution of student who passed project 6

VII. Summary

We have taken an initial step towards identifying at-risk students in a SPOC, which could help instructors design an early warning system. Several prediction models are compared, with SVM preferred due to its good performance. Based on the predicted probabilities, we could help instructors understand students' progress in the course.

The noteworthy accomplishments of our study when compared to other studies including:

- We extracted variable from the click stream logging data and then generate complex features which explain the learners' study behavior, especially describe the learners' study habits.
- We attributed SVM model to those variables as we achieve AUC in the range of 0.62-0.83 for one week ahead.
- Using the most important features in the prediction model, we form an early warning system and a five-dimension assessment scheme. This evaluation model could used in the next iteration as an excitation during the course progress each week.

In Fall 2017 semester, we will deploy our predictive models and do subsequent interventions to at-risk student in the course. Furthermore, we will take more attention to why a student is failing, and what strategies make others' success in a SPOC or on-campus course.

Acknowledgment

Our thanks to the supports from the Teaching Research Funding in Honors College of Beihang University (2017) and China Scholarship Council (No.201406025114).

References

[1] Eric Grimson, John Guttag, and Ana Bell. 2013. "Introduction to Computer Science and Programming using Python", DOI=

https://www.edx.org/course/introduction-computer-science-mitx-6-00-1x-9.

[2] Graham T. Allison, Jr., David E. Sanger, and Derek S. Reveron. 2013. "Central challenges of American National Security, Strategy, and the Press", DOI= https://www.edx.org/course/central-challenges-american-national-harvardx-hks211-2x#.VKzajCuUdwA.

[3] Armando Fox. 2013. "Software Engineering curriculum technology transfer: Lessons learned from MOOCs and SPOCs". *Viewpoint column in Communications of the ACM* 56(12), Dec. 2013.

[4] Wan Han, Gao Xiaopeng, and Liu Qian. 2016. "Hybrid teaching mode for laboratory-based remote education of Computer Structure Course". *The 46th annual Frontiers in Education (FIE) Conference*. Oct. 12-15, 2016 Erie, Pennsylvania, USA, 1-8.

[5] "Events in the Tracking Logs", EdX Research Guide. DOI= http://edx.readthedocs.io/projects/devdata/en/latest/internal_data_formats/tracking_logs.html.

[6] Bydžovská, H. 2016. "A comparative analysis of techniques for predicting student performance". In *Proceedings of the 9th International Conference on Educational Data Mining* (June 29 - July 2, 2016, Raleigh, NC, USA), 306-311.

[7] Wang, F., and Chen, L. 2016. "A nonlinear state space model for identifying At-Risk students in open online courses". In *Proceedings of the 9th International Conference on Educational Data Mining* (June 29 - July 2, 2016, Raleigh, NC, USA), 527-532.

[8] Stapel, M., Zheng, Z., and Pinkwart, N. 2016. "An ensemble method to predict student performance in an online math learning environment". In *Proceedings of the 9th International Conference on Educational Data Mining* (June 29 - July 2, 2016, Raleigh, NC, USA), 231-238.

[9] Elbadrawy, A., Studham, R. S., and Karypis, G. (2015, March). "Collaborative multi-regression models for predicting students' performance in course activities". In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge* (Marist College, Poughkeepsie, NY, March 16-20, 2015). LAK '15. ACM New York, NY, USA, 103-107. DOI= http://dx.doi.org/10.1145/2723576.2723590.

[10] Halawa, S., Greene, D., and Mitchell, J. 2014. "Dropout prediction in MOOCs using learner activity features". In *eLearning Papers, vol. 37, issue March (2014)*, ISSN: 1887-1542, 3-12

[11] He, J., Bailey, J., Rubinstein, B. I., & Zhang, R. 2015. "Identifying At-Risk students in massive open online courses". In *Proceeding of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 1749-1755.

[12] Srilekshmi, M., Sindhumol, S., Chatterjee, S., and Bijlani, K. 2016. "Learning analytics to identify students At-risk in MOOCs". In *2016 IEEE Eighth International Conference on Technology for Education (T4E)* (2 - 4 Dec. 2016), IEEE, 345 E 47TH ST, NEW YORK, NY 10017 USA, 194-199. DOI= https://doi.org/10.1109/T4E.2016.048.

[13] Kloft, M., Stiehler, F., Zheng, Z., and Pinkwart, N. 2014. "Predicting MOOC dropout over weeks using machine learning methods". In *Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs*, 60-65. DOI= http://dx.doi.org/10.3115/v1/W14-4111.

[14] Ren, Z., Rangwala, H., and Johri, A. 2016. "Predicting performance on MOOC assessments using Multi-Regression models". *arXiv preprint arXiv:1605.02269*.

[15] Conijn, R., Snijders, C., Kleingeld, A., and Matzat, U. 2016. "Predicting student performance from LMS data: A comparison of 17 blended courses using Moodle LMS". In *IEEE Transactions on Learning Technologies* (12 October 2016), IEEE, 345 E 47TH ST, NEW YORK, NY 10017 USA, 1-1, DOI= https://doi.org/10.1109/TLT.2016.2616312.

[16] Zhou, Q., Zheng, Y., and Mou, C. 2015. "Predicting students' performance of an offline course from their online behaviors". In *2015 Fifth International Conference on Digital Information and Communication Technology and its Applications (DICTAP)* (29 April - 1 May 2015), IEEE, 345 E 47TH ST, NEW YORK, NY 10017 USA, 70-73. DOI= https://doi.org/10.1109/DICTAP.2015.7113173.

[17] Fei, M., and Yeung, D. Y. 2015. "Temporal models for predicting student dropout in massive open online courses". In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)* (14 - 17 Nov. 2015), IEEE, 345 E 47TH ST, NEW YORK, NY 10017 USA, 256-263. DOI= https://doi.org/10.1109/ICDMW.2015.174.

[18] Colin Taylor, Kalyan V., and Una-May O., 2014. "Likely to stop? Predicting stopout in Massive Open Online Courses". DOI= http://arxiv.org/pdf/1408.3382v1.pdf ..