Personalized Behavior Recommendation: A Case Study of Applicability to 13 Courses on edX

Conference Paper · July 2017

DOI: 10.1145/3099023.3099038

2 authors, including:

Zachary A Pardos
University of California, Berkeley
63 PUBLICATIONS **692** CITATIONS

SEE PROFILE

Personalized Behavior Recommendation: A case study of applicability to 13 courses on edX

Steven Tang
UC Berkeley
Berkeley, CA
steventang@berkeley.edu

Zachary A. Pardos
UC Berkeley
Berkeley, CA
zp@berkeley.edu

ABSTRACT
Individualized and personalized learning has taken on different forms in the context of digital learning environments. In intelligent tutoring systems, individualization is focused on estimation of the cognitive mastery of the student and the speed at which the student progresses through the material is conditioned on her individual rate of mastery. In prior work, a recommendation framework based on learner behaviors, rather than learner's cognitive abilities, was proposed and developed. This framework trained a behavior model on millions of previous student actions in order to estimate how a future learner might behave. This behavior model can incorporate the amount of time spent on each course page, such that the model can take into account a learner's previous behaviors and provide a specific course page recommendation to where the learner may want to go next where they can be expected to spend a significant amount of time on. We stipulate that this approach touches on factors more aligned with personalization, since the prediction of behavior is an aggregation of the student's cognitive abilities, affective state, and preferences. This model was applied to a hand-picked pair of MOOC offerings where model results were expected to be favorable. In this paper, we investigate the suitability of this behavioral prediction approach by applying it to an expanded set of 13 UC Berkeley MOOCs run on the edX platform. Preliminary results from applying the time-augmented Recurrent Neural Network (RNN) based behavior model approach are presented and compared to baseline models. These findings contribute to the discussion of when and in what context this form of collaborative based personalized recommendation is appropriate in MOOCs.

Author Keywords
Adaptivity; Personalization; MOOC; RNN; Behavioral modeling; Navigational efficiency

INTRODUCTION
Digital learning environments have experienced tremendous growth in use over the past several decades. In particular, modeling of cognitive knowledge, skills, and abilities allows Intelligent Tutoring Systems (ITS) to individualize to a student's particular pace of cognitive mastery of the material. Individualization to a student's demonstrated abilities also manifests itself in computer adaptive tests, aimed at increasing the efficiency, accuracy, and precision of high-stakes summative assessments. Massive Open Online Courses (MOOC), a relative newcomer to the landscape of digital learning environments, are promising candidates for student modeling given the high volume of event log data collected by these popular platforms. Like most material in post-secondary education, MOOCs are largely devoid of the kind of knowledge, skills, and abilities tagged to problems in ITS. Why no tagging has emerged is debatable. One argument has been that these college level courses are of a more conceptual nature than the material found in the ITS domains of mostly K-12 STEM, and thus do not lend themselves well to theories of fine-grained atomic skill acquisition. There are also far fewer assessments in the design of a typical MOOC than in ITS, thereby limiting the granularity of skills that can be assessed with precision. Other reasons for the lack of tagging are less principled, including the digitization of existing residential courses simply being the path of least resistance. No matter the cause for differences in pedagogical approach, billions of events have been logged by learners utilizing MOOCs. Given the open navigational interface of the MOOC, we see the mining of these logs as an opportunity to learn pathways learners are taking through the courseware which are a function of their cognitive processing of the material, affect, and personal preferences.

This paper applies a behavior modeling approach that utilizes Recurrent Neural Networks (RNNs) to learn patterns of student navigational behaviors in MOOCs. These models can then be used as part of a personalized recommendation framework that seeks to create more efficient navigations for the learner, with the potential learning benefit being a reduction in time on task to achieve the same performance goals (e.g. certificate / grade) or possibly even an increase in performance without increasing time on task. This paper helps investigate the extent to which such a modeling approach generalizes to different types of MOOCs. Additionally, this paper continues the research conversation on design considerations for personalized recommendation in MOOC contexts. The model accepts a sequence of navigation behaviors, including the amount of time spent on each page, and outputs a probability distribution over the next possible pages a student might navigate to and how much time she is expected to spend on that page. By allowing a learner to jump to the page they are predicted to spend the most time on next, time spent by the user searching for the relevant page could be saved.

RELATED WORK
Personalized adaptation in learning contexts has been researched from a variety of angles. PERSEUS [1] was a framework that provided social navigation support, topic-based navigation support, concept-based navigation support, and adaptive recommendation in educational hypermedia systems. ELM-ART [2] was one of the first Web-based Intelligent Educational systems that offered a combination of Intelligent Tutoring and Adaptive Hypermedia technologies. In this paper, a collaborative filtering approach is used to determine potentially useful resources to recommend to students. Behavioral states of the current student are compared to the abstract behavioral state learned from many students in previous versions of the course. An alternative approach towards recommendation is content-based recommendation [3] [4]. In the content-based approach, recommenders seek to infer relationships between resources by their content. In the case of MOOCs, this could be video transcripts matched to text of an assessment problem to determine how related these two elements were. This is a viable approach towards recommendation in MOOC contexts as well;

however, content-based recommendation is a topic for future work as we explore the utility of the collaborative based method.

Sequential modeling is required to recognize patterns in navigational behaviors and how past navigations might affect future navigation predictions. Several sequential models could be suitable for this task. In other work, tensor factorization was used to model the learning process for students in order to predict future student performance [5]. Such an approach might also be suitable for behavior prediction, but is not explored in this paper.

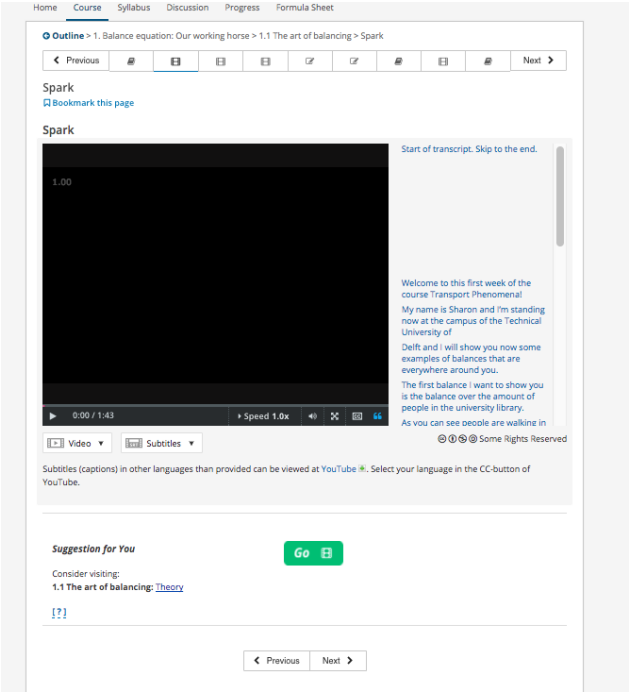

Figure 1 Screenshot of Recommendation Interface (bottom of page) in a live 2017 edX MOOC

The work presented in this paper continues work where behavior in a single MOOC was modeled using a Recurrent Neural Network [6] at a level of granularity where actions such as pausing videos, changing video lecture speed, viewing and answering problems, and navigating through the course page were included as behaviors to be modeled. In that work, the RNN-based Long Short-Term Memory models proved to be successful in capturing signal better than n-gram and course structure baselines. This type of RNN-based model approach was then additionally used as part of a live implementation of a recommendation system [7] deployed in a 2016 edX MOOC. From that live implementation, a student's behaviors during the 2016 offering of the MOOC were modeled using a predictive model of behaviors from the 2015 offering of that same course. In that work, actions were modeled at the granularity of course pages, rather than the more specific actions such as changing video speed or pausing videos. Thus, the granularity at which the behaviors were modeled was slightly different compared to the initial work. Once again, the LSTM and RNN based models found better predictive accuracy than n-gram and course syllabus baselines. The recommendation framework produced a personalized and live recommendation to the student, indicating that the student is likely to spend a significant amount of time on this recommended course page. Figure 1 shows what the student saw on their course page. The recommendation framework consisted of a JavaScript snippet automatically embedded at the bottom of the edX MOOC web page. Note the presence of a large green Go button. The URL that this Go button leads to was informed by the underlying RNN-based behavior model, where the current student's behaviors were fed as input. This button would take the student directly to the page they were predicted to spend over 10 seconds on. In the cases where this suggestion was not simply the next page, this direct linking is anticipated to save the learner time.

In this paper, the RNN based behavior model is applied to a wider selection of MOOCs and preliminary predictive results are reported. This type of investigation will help inform when the behavior-based personalization framework for MOOCs may be beneficially applied and when conditions are not conducive to this approach.

When considering personalized recommendation for a MOOC, several important design choices must be considered. For some courses, it may be the case that a navigational behavior model can substantially increase the efficiency with which a student works through a course due to possibly complex course design with a high degree of referencing of earlier material in the assessments. Other courses, however, may not be well suited for navigation recommendation, but instead may have a diverse selection of content well suited towards content-based recommendation. How and what to recommend content in MOOCs is an open and ongoing research task. This paper specifically addresses how well a navigational behavior model, based on recurrent neural networks, predicts student navigations in a variety of courses. It is expected that some courses will benefit more strongly from behavior modeling compared to others. Testing other types of models for different types of MOOCs remains an open research task.

METHODS

The input to the RNN model consists of the history of courseware URLs that the student has visited. The output of the model consists of a probability distribution over all possible courseware URLs. Thus, the model outputs a probability for every possible next courseware URL navigation, and these probabilities sum to 1.

The model was "time-augmented", whereby the input sequence also included the amount of time spent on each page, estimated by the difference between the timestamps for that page and the next page (clipped to 30 minutes, max). The output probability distribution was also augmented with time, where the label of the next page accessed included the amount of time spent on that page. Time is represented as a concatenation of the index representing the unique pages with one of four time duration categories.

The underlying mechanics of the behavior model are realized through a Recurrent Neural Network (RNN) approach. RNNs [8] were chosen because their sequential model topology lent itself nicely to our navigational sequence prediction task. RNNs have shown recent successes in similarly structured modeling tasks in the domain of natural language [9] and have been previously applied to a math tutoring setting to predict the correctness of responses to skills [10]. RNNs, in theory, are able to model time dependent relationships between events in arbitrarily long sequences without the need for manual feature engineering.

In this paper, accuracy is measured as "recall @ 1," meaning that a prediction is considered correct if the highest probability course page predicted in the output (regardless of time category) is the actual next course page visited by the student. This style of sequential modeling, whereby the input is a long sequence of previous elements and the output is a probability distribution over

all possible elements, is analogous to the approach used in language modeling [11].

An RNN uses all the same components of a simple feed-forward neural network, consisting of an input layer, an arbitrary number of hidden layers, and an output layer. The difference is that this structure is repeated for every time slice of the model, dictated by the maximum length sequence among the learners. The nodes in the hidden layer(s) of the network are all-connected to the hidden nodes in the hidden layer of the next time slice of the model. The weights of the network, including the weights associated with the between time slice connections, are shared in every time slice of the model.

Formally, RNNs maintain a latent, continuous state $\boldsymbol{h_t}$. The RNN model is parameterized by the input weight matrix $\boldsymbol{W_x}$, recurrent weight matrix $\boldsymbol{W_h}$, initial state $\boldsymbol{h_0}$, and output matrix $\boldsymbol{W_y}$. $\boldsymbol{b_h}$ and $\boldsymbol{b_y}$ are biases for the latent and output units. σ and ***tanh*** are sigmoidal squashing functions that compress input, representing the sigmoid and hyperbolic tangent functions respectively.

$$\boldsymbol{h_t = tanh(W_x x_t + W_h h_{t-1} + b_h)}$$
$$\boldsymbol{y_t = \sigma(W_y h_t + b_y)}$$

Long Short-Term Memory (LSTM) [11] is a popular variant of the RNN, whereby the hidden memory unit is augmented with additional gating logic that helps the model learn about long range dependencies. This gating logic learns when to retain and when to forget information in the latent state. Each hidden state $\boldsymbol{h_t}$ is replaced by an LSTM cell unit that maintains the additional gating parameters.

$$\boldsymbol{f_t = \sigma(Wf_x x_t + W_{fh} h_{t-1} + b_f)}$$
$$\boldsymbol{i_t = \sigma(W_{ix} x_t + W_{ih} h_{t-1} + b_i)}$$
$$\boldsymbol{C'_t = tanh(W_{Cx} x_t + W_{Ch} h_{t-1} + b_C)}$$
$$\boldsymbol{C_t = f_t \times C_{t-1} + i_t \times C'_t}$$
$$\boldsymbol{o_t = \sigma(W_{ox} x_t + W_{oh} h_{t-1} + b_o)}$$
$$\boldsymbol{h_t = o_t \times tanh(C_t)}$$

$\boldsymbol{f_i}$, $\boldsymbol{i_t}$, and $\boldsymbol{o_t}$ represent the gating mechanisms used by the LSTM to determine when to forget, input, and output data from the cell state, $\boldsymbol{C_t}$. $\boldsymbol{C'_t}$ represents an intermediary candidate cell state that is gated to update the next cell state. $\boldsymbol{h_t}$ and $\boldsymbol{C_t}$ are the memory components that are used to propagate information between timesteps and inputs, thus passing information to future predictions in the sequence.

LSTM models contain several hyperparameters that affect how the model performs on training data [12]. For these preliminary results, the number of LSTM layers, the number of nodes in each layer, and the optimization algorithm were varied. LSTM layers varied between 1, 2, or 3, the number of nodes per layer were 32, 64, 128, or 256, and RMSProp and ADAM were tried as optimization methods [13]. This indicates that a total of 24 different hyperparameter sets were fit per dataset. For each MOOC dataset, 30% of students' actions were held out as a test set. Of the 70% of the student data used as the training set, 10% was held out as a hill-climbing set. When the calculated loss on the hill-climbing set did not improve for 3 consecutive epochs, training was stopped and the learned weights were saved; this process is known as "early-stopping," as a method to account for over-fitting. The set of hyperparameters that had the highest accuracy on the hill-climbing set was then saved, and the model was then tested on the 30% held-out test set. Thus, of the 24 hyperparameter sets, only the best performing set according to hill-climbing accuracy was tested on the 30% held out set. This process was repeated for both the vanilla LSTM approach and the time-augmented LSTM approach [7].

BASELINE MODELS

Two baseline models are used. The first is the "Next Syllabus URL" model, which always predicts that the student will go to the next URL in the course structure. This is equivalent to the student always clicking the Next button rendered in the browser. The second baseline is the "Next Most Common URL" model, which predicts the course page that most commonly followed a particular URL in the 70% of training data. This is equivalent to a 2-gram approach [14]. The syllabus model is a useful baseline since it relates directly to the implied structure of the course set by the course instructor. The next most common model is a simple yet intuitive heuristic that accounts for very common deviations in navigation behavior from the syllabus model.

DATASETS

For this preliminary work, 13 UC Berkeley MOOC datasets from courses administered in late 2015 to 2016 from the edX platform were analyzed. These datasets range from as few as 204 students to as many as 13,998 students who answered at least one quiz question. The number of navigational actions recorded ranges from as few as 8,421 to as many as 1,268,706 actions. Note that this is a filtered subset of actions, whereby a process of "duplicated navigations" were removed. Thus, if a student performs a navigation to the same page, either through refreshing the page or navigating to the same page via the browser navigation buttons, those navigations were removed. Each student sequence of navigations was also truncated to the length of the 99.5 percentile length for each particular course's dataset. This helps alleviates training time by allowing for a smaller length of vector as input to the LSTM model structure. Additionally, there exists the possibility of very long student sequences that are produced by web crawlers or other types of bots, and this helps to mitigate some of those concerns.

Table 1 Dataset Sizes. Column abbreviations stand for Number of (N)avigational behaviors, Number of (S)tudents in the dataset, Number of unique (C)ourse pages that exist in the syllabus, and Measure of (E)ntropy.

BerkeleyX Course	N	S	C	E
BJC.3x 1T2016	23435	231	81	0.42
BJC.4x 1T2016	20004	204	115	0.55
ColWri2.2x 1T2016	792577	13998	54	0.36
ColWri2.3x 1T2016	91569	1570	46	0.21
ColWri.3.10 1T2016	25558	814	24	0.46
ColWri.3.11 1T2016	18358	555	25	0.48
ColWri.3.12x 1T2016	14859	416	28	0.55
ColWri3.1x 2_3T2015	33852	767	30	0.65
ColWri3.2x 2T2016	8421	210	27	0.42
ColWri3.3x 2T2016	12729	226	33	0.54
EE40LX 2T2015	1268706	9605	287	0.61
Fin101x 1T2016	65466	1534	113	0.62
Policy01x 1T2016	50038	637	91	1.12

Table 1 displays four columns: (N) Number of navigational behaviors, (S) Number of students in the dataset, (C) Number of unique course pages that exist in the syllabus, (E) Measure of

entropy [15], a measure of variation in navigational pathways generated by treating student paths through a course as a Markov Chain and then computing the entropy of the transition probability matrix. A higher amount of entropy indicates a larger amount of variation in navigation. The Course column corresponds to the recorded course ID logged by edX. Each of these courses have the following proper titles; BJC refers to the Beauty and Joy of Computing (CS principles), parts 3 and 4. ColWri is short for College Writing, and each of the courses focus either on a set of novel(s) or different writing styles. EE40LX refers to an Electrical Engineering course pertaining to Electronic Interfaces. Fin101x refers to a finance course about how to save money. Policy01x refers to a public policy course relating to UC Berkeley's "Eightfold Path", whereby the course discusses solving public policy problems.

PRELIMINARY RESULTS

The first step in selecting a behavioral model for a course was to tune the hyper parameters of the model and Table 2 shows the set of hyperparameter values that performed best on the 10% hill-climbing set. The hyperparameters for both the vanilla LSTM and the Time-Augmented LSTM (TLSTM) models are shown. The hyperparameters are displayed in the format (L, N, E, O), standing for Layers, Nodes, Epochs, and Optimizer. Layers represents the number of LSTM layers in the model, Nodes represents the number of nodes per LSTM layer, Epochs represents the epoch at which the best hill-climbing accuracy was produced, and Optimizer represents the best optimizer, which is either Adam or RMSprop, denoted as A and R in the table respectively.

Table 2 Best Hyperparameter Values According to Hill Climbing Accuracy

BerkeleyX Course	LSTM	TLSTM
BJC.3x 1T2016	1, 32, 81, A	1, 64, 41, A
BJC.4x 1T2016	1, 32, 50, R	1, 32, 69, R
ColWri2.2x 1T2016	1, 256, 9, A	1, 256, 10, A
ColWri2.3x 1T2016	3, 64, 35, R	1, 64, 42, A
ColWri.3.10 1T2016	2, 256, 37, A	1, 64, 31, A
ColWri.3.11 1T2016	2, 64, 31, A	2, 128, 24, A
ColWri.3.12x 1T2016	1, 256, 16, A	2, 32, 42, R
ColWri3.1x 2_3T2015	1, 64, 48, A	2, 64, 36, R
ColWri3.2x 2T2016	2, 32, 57, A	1, 64, 33, A
ColWri3.3x 2T2016	2, 128, 46, A	1, 32, 37, R
EE40LX 2T2015	2, 256, 33, A	1, 256, 15, R
Fin101x 1T2016	2, 64, 25, A	2, 64, 53, A
Policy01x 1T2016	1, 256, 20, A	3, 128, 60, A

There did not exist a single set of hyperparameters that performed universally best across all datasets. In 19 of the 26 dataset+model formulation combinations, the Adam optimizer was found in the best performing hyperparameter set. In 14 of the 26 these combinations, 1 layer models performed best, while the remaining 12 had either 2 or 3 layers. The number of nodes and the number of epochs varies throughout. These preliminary results indicate a need for a separate grid search each time an RNN-based model is utilized for MOOC behavior recommendation.

Table 3 presents results of the different models applied to the 13 BerkeleyX MOOCs. The 4 models shown are Course Syllabus (S), Next Most Common (NMC), Vanilla LSTM (LSTM), and Time-Augmented LSTM (TLSTM). The accuracy is the model's average performance across all students in the 30% held-out test set of each dataset. Thus, the NMC and LSTM models were trained on the same 70% of data and tested on the same 30% test set. The syllabus model is not a trained model, so its accuracy depends purely on the test set. The last 3 columns of the table show the percentage increase in performance of each model over the Course Syllabus baseline, with the highest percentage increase bolded. For example, **% NMC / S** referms to the absolute accuracy increase of the NMC model over the S model converted to a percentage of the accuracy of the S model. This gives a sense of how much the models are improving relative to the accuracy of the S model

In 10 out of the 13 courses tested, the TLSTM model has the best performance of the four tested models. In 2 of the remaining 3, the vanilla LSTM model performed the best. In one course, the NMC model shows the best performance. The difference in model performance could be due to a variety of factors. There was clear variation in the relative performance benefit of the TLSTM model over the S model, compared to the benefit of the NMC over the S model. For example, for **ColWri2.2x 1T2016**, NMC outperformed S by just 0.04%, while the TLSTM outperformed S by 9.49%. For **BJC.4X 1T2016**, the relative improvement of NMC / S outperformed the TLSTM / S again. These preliminary results indicated that in the general case, the TLSTM approach was, as expected, the strongest performing model, but for some types of datasets, the TLSTM was not generalizing well to the test set.

When comparing model performance, it is useful to see to what extent different models agree or disagree given the same test data. Figure 2 compares how often the Syllabus and the TLSTM models agreed on test set prediction. Each bar represents 0 to 100% correctness, where 100% correctness comprises the set of test set predictions that at least one of the Syllabus or the TLSTM model predicted correctly. Thus, in this graph, each bar examines to what extent both models either both made the same correct prediction and to what extent one model made a correct prediction while the other model made an incorrect prediction. The blue part of each bar represents the percentage of correct predictions that were correctly predicted by both models. The orange component represents the proportion of correct predictions that the Syllabus model predicted correctly, but the TLSTM model predicted incorrectly. The gray component represents the proportion of predictions that the TLSTM model predicted correctly, but the Syllabus model predicted incorrectly. Note that these are based off the raw number of correct and incorrect predictions, which is slightly different from the accuracy metric presented in Table 3, where accuracies are averaged across students rather than by raw correct and incorrect predictions. This difference is due to the fact that students have varying length sequences. Thus, when looking at accuracy among all possible predictions, sequences that are longer than average will affect the average more significantly, whereas when accuracy is averaged across all students, the importance of longer sequences is weighted similarly to shorter length sequences. This figure helps provide a sense of the variation in how often the Syllabus and TLSTM models agree and disagree with one another. It is clear from the figure that the TLSTM generates a larger proportion of

Table 3 Preliminary Results. Column abbreviations stand for (S)yllabus, Next Most Common (NMC), and (T)ime (LSTM)

BerkeleyX Course	S	NMC	LSTM	TLSTM	% NMC / S	% LSTM / S	% TLSTM / S
BJC.3x 1T2016	0.578	**0.591**	0.577	0.588	**2.22**	-0.15	1.76
BJC.4x 1T2016	0.530	0.536	0.523	**0.543**	1.22	-1.33	**2.49**
ColWri2.2x 1T2016	0.611	0.612	0.648	**0.670**	0.04	5.91	**9.49**
ColWri2.3x 1T2016	0.631	0.631	0.659	**0.681**	-0.04	4.4	**7.85**
ColWri.3.10 1T2016	0.510	0.578	0.576	**0.593**	13.27	12.98	**16.25**
ColWri.3.11 1T2016	0.487	0.555	0.563	**0.569**	13.79	15.47	**16.83**
ColWri.3.12x 1T2016	0.482	0.536	0.537	**0.542**	11.22	11.37	**12.47**
ColWri3.1x 2_3T2015	0.509	0.530	0.557	**0.567**	4.1	9.44	**11.52**
ColWri3.2x 2T2016	0.501	0.506	**0.525**	0.516	0.85	**4.64**	2.94
ColWri3.3x 2T2016	0.477	0.515	**0.543**	0.523	7.96	**13.98**	9.76
EE40LX 2T2015	0.662	0.665	0.694	**0.698**	0.4	4.82	**5.4**
Fin101x 1T2016	0.695	0.696	0.700	**0.706**	0.04	0.73	**1.58**
Policy01x 1T2016	0.545	0.571	0.590	**0.591**	4.65	8.19	**8.32**

uniquely correct predictions. Additionally, the degree to which the TLSTM outperforms the Syllabus model varies between courses.

Using the four course attributes from Table 1 as features in a linear regression to the percent improvement of TLSTM over the Syllabus model from Table 3, the following model resulted (after normalizing all terms between 0 and 1): $y = 0.652 + 2.49 \times \text{NumBehaviors} - 1.37 \times \text{Students} - 2.19 \times \text{Pages} + 0.54 \times \text{Entropy}$.

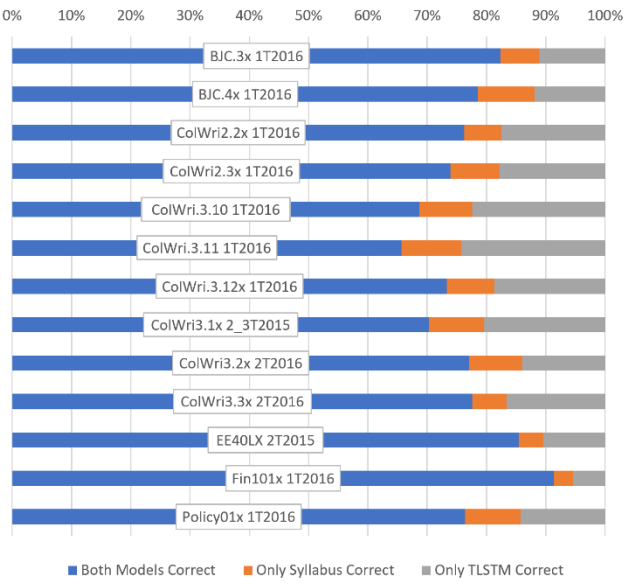

Figure 2 Model Correctness as Percentages of Either Model Predicting a Specific Navigation Correctly

DISCUSSION AND FUTURE WORK

The preliminary results here indicate that the TLSTM model generally performed above the Syllabus and NMC baselines. High volume of data (NumBehaviors) and high variation in pathways (Entropy) contributed most positively to the TLSTM's predictive performance over the course syllabus baseline, according to a linear regression.

Future work is planned to expand this analysis to a multitude of courses, including many courses not administered by BerkeleyX. We also intend to include a greater number of course attributes in our regression analysis, including number of assessment problems in the course, certification rate, level of instructor or GSI involvement in the forums, among other attributes. There is potentially a case to be made for this personalized guidance being most helpful in low-touch self-paced courses, which we will explore further.

When should data be used to adapt the courseware versus augment the course with recommendations? How do we work towards cognitive based adaptivity from observational data while avoiding false causal inferences? What is the learner seeking and how should a guidance framework balance meeting the wants and needs of the learner with the values and expectations of the instructor? We pose these open questions of import to the community for further discussion.

Acknowledgement

This work was supported by the National Science Foundation (NSF Award #1547055).

REFERENCES

[1] M. Yudelson, "Providing Service-based Personalization in an Adaptive Hypermedia System," University of Pittsburgh, 2010.

[2] G. Weber and P. Brusilovsky, "ELM-ART – An Interactive and Intelligent Web-Based Electronic Textbook," *Int J Artif Intell Educ*, vol. 26, no. 72, 2016.

[3] M. Pazzani and D. Billsus, "Content-based recommendation systems," *The adaptive web*, pp. 325-341, 2007.

[4] P. Brusilovsky and E. Millán, "User models for adaptive hypermedia and adaptive educational systems," *The adaptive web*, 2007.

[5] S. Sahebi, Y. -R. Lin and P. Brusilovsky, "Tensor Factorization for Student Modeling and Performance Prediction in Unstructured Domain," in *Proceedings of the 9th International Conference on Educational Data Mining*, Raleigh, NC, 2016.

[6] S. Tang, J. C. Peterson and Z. A. Pardos, "Predictive Modeling of Student Behavior Using Granular Large Scale Action Data from a MOOC," *Handbook of Learning Analytics and Educational Data Mining*, 2017.

[7] Z. A. Pardos, S. Tang, D. Davis and C. V. Le, "Enabling Real-Time Adaptivity in MOOCs with a Personalized Next-Step Recommendation Framework," *Learning at Scale*, 2017.

[8] T. Mikolov, M. Karafiát, L. Burget and S. Khudanpur, "Recurrent neural network based language model," *Interspeech*, 2010.

[9] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint*, 2013.

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

[11] M. Sundermeyer, R. Schlüter and H. Ney, "LSTM Neural Networks for Language Modeling," *Interspeech*, 2012.

[12] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE transactions on neural networks and learning systems*, 2016.

[13] K. Diederik and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[14] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. Pietra and J. C. Lai, "Class-based n-gram models of natural language," *Computational linguistics*, vol. 18, no. 4, pp. 467-479, 1992.

[15] S. Reddy, I. Labutov and T. Joachims, "Latent skill embedding for personalized lesson sequence recommendation," in *CoRR, abs/1602.07029*, 2016.

[16] S. Tang, J. C. Peterson and Z. A. Pardos, "Modelling Student Behavior using Granular Large Scale Action Data from a MOOC.," arXiv preprint arXiv:1608.04789, 2016.