Adaptive Engine for MOOC

Ilia Rushkin
Cambridge, USA

ABSTRACT
UPDATED—April 22, 2017. Description of an adaptive recommendation engine for serving assessment questions in a MOOC. Prototype in R exists.

1. INTRODUCTION

We would like to create a simple adaptive recommendation engine for a MOOC, capable of deciding what url to serve to a user next based on the user's history and the information about the url's content, provided by an SME. Primarily, we are interested in serving assessment items (below we call them questions), but instructional materials are also possible. These can be instructional webpages or videos intended to be mixed with assessment items ("if a student has trouble with that question, let them read this" etc.)

We use a variety of Bayesian Knowledge Tracing (BKT) model to estimate the students' state. What makes our situation special is that, as we learned from the adaptive pilot and just generally from seeing MOOCs,

1) Questions in the course differ widely in nature, and in particular in difficulty. Thus, we cannot assign the same values of guess, slip and transit probabilities to them, even if they are all tagged with the same learning objective.

2) Tagging is complicated: often a question is tagged with several learning objectives (aka "skills", "knowledge components"), possibly with varying degree of relevance.

3) In a self-paced MOOC environment, there is a need for a causal structure in the learning objectives: we should not serve to a user items tagged with a learning objective, if the user has shown lack of knowledge of other learning objectives that are pre-requisite to that one. In the simplest case, it can be dictated by a simple ordered list (the natural order of learning the content of the course), but it could also be a detailed graph of pre-requisite relationships among learning objectives.

4) In a MOOC, the number of students is high, so we can afford to define a model with a large number of parameters and optimize them based on the student interaction data.

2. RELATED LITERATURE

TBD

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

3. MODEL DESCRIPTION

Knowledge tracing part

The content knowledge of the course is represented as a list of N LOs, short for "learning objectives" with pre-requisite relationships traced among them. These relations will be involved in the recommendation part of the engine, but not in knowledge tracing.

For each question q ($q = 1, 2, ...Q$) of the course, we define the relevance of each LO i, denoted k_{qi} and on the scale from 0 to 1. Thus, we form a $Q \times N$ matrix k, each row of which is for a question and each column is for an LO. This matrix is the result of tagging questions with LOs.

We assume that the mastery of each LO by each course user is a binary latent variable – the user either has learned it or not – and we update the mastery matrix p, where the element $p_u i$ is the currently estimated probability that the user u has the mastery of the LO i. We could define the mastery threshold $p^* \in [0,1]$, and if $p_{ui} \geq p^*$, we say that the mastery of i by the user u is sufficiently certain and no longer needs verification. We could initialize the mastery probability matrix $p = p^{(0)}$ (students' prior knowledge), after which, when the user has been served a question. When the user submits an answer to the question, it gets a correctness value (score) $C_q^{(u)} \in [0,1]$ and we update the mastery probability of each LO (i.e. this user's row of the matrix p).

The Bayesian updating is easier to write in terms of the logarithmic odds rather than the probability p:

$$L_{ui} = \log\frac{p_{ui}}{1-p_{ui}}, \quad L_{ui}^{(0)} = \log\frac{p_{ui}^{(0)}}{1-p_{ui}^{(0)}}, \quad L^* = \log\frac{p^*}{1-p^*} \tag{1}$$

So we will translate the transit, guess and slip probabilities into odds:

$$o_{qi}^{trans} = \frac{p_{qi}^{trans}}{1-p_{qi}^{trans}}, \quad o_{qi}^{guess} = \frac{p_{qi}^{guess}}{1-p_{qi}^{guess}}, \quad o_{qi}^{slip} = \frac{p_{qi}^{slip}}{1-p_{qi}^{slip}}, \tag{2}$$

Then the update procedure requires defining the logarithms of likelihood ratios for the case of incorrect (0) and correct (1) answer:

$$x_{qi}^0 = \log\frac{p_{qi}^{slip}}{1-p_{qi}^{guess}}, \quad x_{qi}^1 = \log\frac{1-p_{qi}^{slip}}{p_{qi}^{guess}} \tag{3}$$

These matrices encode the relevance of a problem q to an LO i. If the problem is irrelevant, the probability of correct or incorrect score should be independent of that LO. This will be the case if $p_{qi}^{slip} = 1 - p_{qi}^{guess}$, in which case $x_{qi}^0 = x_{qi}^1 =$

0. We propose to define the relevance matrix, used in the recommendation part of the engine, as:

$$k_{qi} = x^1_{qi} - x^0_{qi} = -\log o^{guess}_{qi} - \log o^{slip}_{qi}, \tag{4}$$

a sum of logarithmic odds of non-guessing and non-slipping. The increment earned by L_{ui} is:

$$x_{qi} = x^0_{qi} + C^{(u)}_q (x^1_{qi} - x^0_{qi}) \tag{5}$$

The posterior odds, with the evidence of the submitted problem, become $L_{ui} \to L_{ui} + x_{qi}$. Additionally, we modify the mastery odds due to transfer of knowledge ($p_{ui} \to p_{ui} + (1 - p_{ui}) p^{trans}_{qi}$), so the full update procedure is:

$$L_{ui} \to \log\left(o^{trans}_{qi} + (o^{trans}_{qi} + 1)\exp(L_{ui} + x_{qi})\right) \tag{6}$$

This is a type of Bayesian Knowledge Tracing, with several modifications. The main one is that we allow for the question to be tagged with multiple LOs, so that the parameters carry the index i. If a problem is tagged with several LOs ($k_{qi} > 0$ for more than one value i), we essentially view the problem as a collection of sub-problems, each tagged with a single LO. The predicted odds $\mathscr{O}^{predict}_{qu}$ correct answer by user u on a question q is found as

$$\mathscr{O}^{predict}_{qu} = \prod_i \frac{e^{L_{ui}}(1 - p^{slip}_{qi}) + p^{guess}_{qi}}{e^{L_{ui}} p^{slip}_{qi} + 1 - p^{guess}_{qi}}, \tag{7}$$

which is to say that we take the ratio of the probability that each sub-problem is answered correctly to the probability that each sub-problem is answered incorrectly (since we must remove from the ensemble the possibilities of correct answer on some but not all sub-problems). The predicted probability of the correct answer to a problem is found from the odds by the standard transformation as: $\mathscr{O}^{predict}_{qu} / (\mathscr{O}^{predict}_{qu} + 1)$.

As written, the update procedure (1-6) readily lends itself to generalization to non-binary scores $C^{(u)}_q$. We may try it instead of dichotomizing the scores in a separate preparatory step.

For terminological simplicity we referred to the content items as questions. However, the model can accomodate instructional items as well, e.g. videos or text. We can adopt a rule that, if an item q is instructional, the outcome of user's interaction with it is always "correct". A way to think of it is to imagine that q includes an assessment part of trivial difficulty. The slip probabilities $p^{slip}_{qi} = 0$, the guess probabilities now have the meaning of the probability of not learning an LO from the item, and so we set them to $p^{guess}_{qi} = 1 - p^{trans}_{qi}$. Also, in case of an instructional item, unlike an assessment item, we may or may not want to mark it as "seen", because it is allowed to serve it to a user multiple times (or perhaps we should impose a limit on how many times it can be served).

If the matrix p or other parameter matrices contain zeros it is possible to encounter indeterminacies of the 0^0 and $log(0)$ types. One way to preclude these is adopt a small cutoff, e.g. we can set $\varepsilon = 10^{-10}$, and make all elements of the parameter matrices $p^{slip}, p^{guess}, p^{transit}$, as well as the initial mastery probability $p^{(0)}$, be in the interval $[\varepsilon, 1-\varepsilon]$. Evidently, if the initial values of the matrix p are all non-zero, the only way zeros can appear is if users work on questions with the slip probability 0 or 1 and give correct/incorrect results contrary to these.

Recommendation part

The pre-requisite relationships among the LOs are naturally visualized as a directed acyclic graph, and is stored as an $N \times N$ matrix w of pre-requisite strengths, $w_i j$ representing the strength of the graph edge from LO j to LO i (j is a pre-requisite for i). We define this strength to be on the scale from 0 to 1. If there are no connections, w a zero matrix.

For each LO and for each user, we can define the pre-requisite readiness:

$$r_{ui} = \sum_{j=1}^{N} w_{ij} \min\left(0, L_{uj} - L^*\right). \tag{8}$$

An element r_{ui} has value 0 if the user has sufficiently mastered all LOs pre-requisite for the LO i, and less than 0 if the matery probabilities for some pre-requisites are not yet certain. If the pre-requisite strength w_{ij} is weaker, it enters r_{ui} with a smaller weight, in a sense allowing less certain mastery of less important pre-requisites. If all the pre-requisites are ascertained, $r_{ui} = 0$, otherwise it is less than 0. We can deviate from this slightly and introducing a forgiveness parameter $r^* \leq 0$, we assume that a user u is sufficiently ready for learning an LO i if $r_{ui} + r^* \geq 0$. If $r^* > 0$, it means that we "forgive" some degree of uncertain knowledge of pre-requisites.

To recommend the next question for a student, we subset the matrix k_{qi} to only those questions (matrix rows) that belong to the current homework and that a user u has not seen yet. Thus, we obtain a user specific matrix $k^{(u)}_{qi}$. We define the non-negative user-specific vectors "question readiness", "question demand", and "question appropriateness" (in terms of difficulty level of the problem $d_q \in [0,1]$):

$$R^{(u)}_q = \sum_{i=1}^{N} k^{(u)}_{qi} \min(0, r_{ui} + r^*) \tag{9}$$

$$D^{(u)}_q = \sum_{i=1}^{N} k^{(u)}_{qi} \max\left(0, L^* - L_{ui}\right) \tag{10}$$

$$A^{(u)}_q = -\sum_{i=1}^{N} k^{(u)}_{qi} \left| L - \log \frac{d_q}{1 - d_q} \right| \tag{11}$$

$$C^{(u)}_q = \sum_{i=1}^{N} k^{(u)}_{qi} k_{q_{last}, i}, \tag{12}$$

where q_{last} is the last item the user saw. To bring them all to comparable scale, we normalize each vector with non-zero range to range 1:

$$R^{(u)}_q \to \frac{R^{(u)}_q}{\max(R^{(u)}_q) - \min(R^{(u)}_q)}, \quad \text{and similar for } D^{(u)}_q, A^{(u)}_q, \tag{13}$$

where it is understood that the renormalization is performed only if the denominator is not 0.

These expressions are invented, they can be modified in many ways. The idea is that we want to maximize some linear combination of readiness, demand and appropriateness. As defined here, $R_q^{(u)} \leq 0, D_q^{(u)} \geq 0, A_q^{(u)} \leq 0$. A larger demand for a question means that this question focuses on those LOs for which the user's probability of mastery is currently low. We introduce a vector of importance weights $V = (V_r, V_d, V_a, V_c)$ (defined up to normalization) which measures the relative importance of each: the next item q to serve should be the one which maximizes the combination $V_r R_q^{(u)} + V_d D_q^{(u)} + V_a A_q^{(u)} + V_c C_q^{(u)}$. The serving stops in two cases:

1) if we exhausted the available questions (the matrix $k^{(u)}$ has no rows)

2) if $D_q^{(u)} = 0$ for all q, which means that the user has reached the mastery threshold p^* on all LOs relevant for the available questions.

4. MODEL PARAMETERS

As described in section 3, the model is governed by the following constants: the mastery threshold p^* (maybe 0.95), the pre-requisite forgiveness parameter r^* (maybe 0.95), the vector of relative importances of readiness/demand/appropriate difficulty V (maybe $V_r = 3, V_d = 1, V_a = 2$) and the regularization cutoff ε (maybe 10^{-10}). In addition, as will be described in section 5, there are threshold for updating BKT parameters: η (maybe 0) and M (maybe 20).

To form the matrix w, we require an SME to produce the pre-requisite relationship graph of learning objectives, indicating the strength of the relations. Realistically, we expect the SMEs to use three distinct values of strength: "none" (the absence of the edge in the graph), "weak" and "strong", and we will then convert these to numeric values adopting some convention (e.g. "none"=0,"weak"=0.5, "strong"=1). How complex a graph the SME produces will vary strongly. But at a minimum, we should have a chain of learning objectives indicating the order in which they should be learned.

To form the matrices $p^{guess}, p^{slip}, p^{trans}$, we require an SME to tag each content item with relevant LOs, indicating the level of relevance of each. Realistically, we expect the SME to tag each item with only a few (possibly just one) LOs, describing their relevance as "weak" or "strong" as measured by their answers to:

1. "How likely is it to solve this problem correctly despite not knowing the LO?" (From here we get the guess probability)

2. "How likely is it to fail this problem despite knowing the LO?" (From here we get the slip probability)

3. "How likely is it that a student who did not know the LO will learn it from working on this problem?" (From here we get the transfer probability)

4. "Difficulty level of this problem" (From here we get the difficulty d_q.

For those LOs that are not listed in SME's tagging, we will set the guess and slip probabilities both to 0.5, and the transfer probability to 0.

Finally, we need to initialize the mastery probability matrix p. At its simplest, we can settle at $p^{(0)} = 0.5$.

Thus, to form the data matrices, we rely partly on SMEs judgement, partly on simple estimates, partly on the choice of the constants used to translate SMEs terms ("weak", "easy", etc) into numbers. For the generalized BKT parameters (matrices p^{slip}, p^{trans}, p^{guess} and $p^{(0)}$) this matters only initially: once we have a substantial amount of student data, we will optimize them (section 5).

5. OPTIMIZATION OF BKT PARAMETERS

[THIS SECTION IS VERY MUCH UNDER DEVELOPMENT. DIFFERENT COMPETING IDEAS.] We will rely on a way to optimize our BKT parameters, inspired by the "empirical probabilities" method of [1].

At some point in time, when we decide to run the optimization, suppose that the items submitted by a user u are $\{q_j^{(u)}\}$ ($j = 1, \ldots J^{(u)}$), indexed in chronological order. Let the correctness of answers be $C_j^{(u)}$. We denote $K_{ij}^{(u)}$ this student's knowledge of an LO i just before submitting the item $q_j^{(u)}$. Assuming that there is no forgetting, the knowledge is a non-decreasing function with values 0 and 1, so it is characterized simply by the position of the unit step: for j from 1 to some n_i knowledge is 0 and from $n_i + 1$ onward it is 1. We find which n_i gives the highest accuracy of predicting correctness from knowledge. The generalized number of errors on predicting the outcome based on mastery of a particular learning objective are:

$$E_i^{(u)}(n) = \left(- \sum_{j=1}^{n} C_j^{(u)} \log o_{q_j,i}^{guess} - \sum_{j=n+1}^{J^{(u)}} (1 - C_j^{(u)}) \log o_{q_j,i}^{slip} \right), \tag{14}$$

where $n \in [0, J^{(u)}]$ and we adopt the convention that if the lower limit of a sum is greater than the upper limit, the sum is 0. We set the knowledge step where it minimizes the errors:

$$n_i = \operatorname{argmin}(E_i^{(u)}), \tag{15}$$

and construct the step-function $K_{ij}^{(u)}$ using it n_i. If there are multiple equal minima, and so multiple n_i, we take the average of the corresponding multiple step-functions (because of this, knowledge may now have fractional value). The resulting $K_{ij}^{(u)}$ is our empirical estimate of the knowledge of all LOs by the user u. Repeat the procedure for each user. Note that, if user's problems are irrelevant for an LO, we will find a steadily growing knowledge of that LO. This is not bad, however, since for each LO we will average only over the users who experienced some relevant problems. Namely, we can define the sets of users

$$\mathcal{U}_i = \{\forall u : \sum_{j=1}^{J^{(u)}} k_{q_j^{(u)},i} > \eta\}, \tag{16}$$

$$\mathscr{U}_{qi} = \{\forall u : \sum_{j=1}^{J^{(u)}} k_{q_j^{(u)},i} \mathbf{1}(q_j^{(u)} = q) > \eta\} \tag{17}$$

where $\eta \geq 0$ is a constant we set as a measure of how much total relevance of a LO is enough for the user to be included into the ensemble for estimating the parameters of that LO. As the simplest choice, $\eta = 0$.

Since the order of items was chronological, $K_{i,1}^{(u')}$ estimates the prior knowledge of concepts by that user. Moreover, we can identify the occasions when slips, guesses or transfers of knowledge occured. Averaging these over the users we get the empirical matrices of prior knowledge, transit, guess and slip probabilities as ratios:

$$P_{u'i}^{(0)} = \frac{\sum_{u \in \mathscr{U}_i} K_{i,1}^{(u)}}{\sum_{u \in \mathscr{U}_i} 1} \tag{18}$$

(same priors for all users u', i.e. all rows of $P^{(0)}$ are identical)

$$P_{qi}^{trans} = \frac{\sum_{u \in \mathscr{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}-1} (1 - K_{ij}^{(u)}) K_{i,j+1}^{(u)} \mathbf{1}(q_j^{(u)} = q) \right)}{\sum_{u \in \mathscr{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}-1} (1 - K_{ij}^{(u)}) \mathbf{1}(q_j^{(u)} = q) \right)} \tag{19}$$

$$P_{qi}^{guess} = \frac{\sum_{u \in \mathscr{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}} (1 - K_{ij}^{(u)}) C_j^{(u)} \mathbf{1}(q_j^{(u)} = q) \right)}{\sum_{u \in \mathscr{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}} (1 - K_{ij}^{(u)}) \mathbf{1}(q_j^{(u)} = q) \right)} \tag{20}$$

$$P_{qi}^{slip} = \frac{\sum_{u \in \mathscr{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}} K_{ij}^{(u)} (1 - C_j^{(u)}) \mathbf{1}(q_j^{(u)} = q) \right)}{\sum_{u \in \mathscr{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}} K_{ij}^{(u)} \mathbf{1}(q_j^{(u)} = q) \right)} \tag{21}$$

Here again, we adopt the convention that if the lower limit of a sum is greater than the upper limit, the sum is 0 (this happens when $J^{(u)}$ is 0 or 1). The value of the denominator in each of these expressions is a measure of how much student information we have for estimating the probability. In case there is no data, the expression becomes a 0/0 indeterminancy. We should not want to update a probability in this case. Moreover, we impose a threshold M (e.g. 20) and say that we will not update a particular probability if the denominator in the corresponding equation is less than M. This is simple to enforce in practice: add to eqq. (18, 19,20,21) the rule that in each of them the denominator less than or equal to M (this roughly counts the number of terms in the denominator sum) should be replaced by 0. After that, any non-numeric (indeterminate or infinite) elements of the matrices $P^{(0)}$, P^{trans}, P^{guess}, P^{slip} should be replaced by the corresponding elements of the matrices $p^{(0)}$, p^{trans}, p^{guess}, p^{slip}.

We should also watch out for guess and slip probabilities of 0.5 and up. If this happens, we will not update with such values.

Now we can update the BKT parameter matrices with the estimates: $p^{(0)} = P^{(0)}$, $p^{trans} = P^{trans}$, $p^{guess} = P^{guess}$, $p^{slip} = P^{slip}$, and also convert probabilities to odds, i.e. $L_{ui}^{(0)} = \log(p_{ui}^{(0)}/(1 - p_{ui}^{(0)})$, and the guess, slip, transit odds via (2).

In addition to that, we should also update some elements of the current mastery odds L. Namely, let us call "pristine" those elements of L, which have never been updated with a non-zero shift via (6). All the pristine elements of should be replaced with the corresponding elements of $L^{(0)}$. This way, the optimized prior LO knowledge values will be used for all users yet to come to the course, but also for the existing users for those learning objectives that they have not yet explored.

TESTING OF THE KNOWLEDGE-TRACING PART

We used the data from the HarvardX course Super-Earths v3 for testing the predictive power of the knowledge-tracing algorithm. The problems in this course tagged with 66 LOs by SMEs. Unfortunately, the number of problems was disproportionately small for this many LOs. 22 of the LOs had only one problem associated with them, 18 LOs had 2 problems, 13 LOs – 3 problems, 8 LOs – 4 problems, 2 LOs – 5 problems, 3 LOs – 7 problems. So we cannot expect much, but we want to at least see that nothing crazy happens when problems are tagged with more than 1 LO. Restricting ourselves to LOs with more than 3 problems (13 LOs) leaves 60 problems, 3 of which are tagged with 2 LOs, with 2,362 users attempting them. We randomly split the users into a training set and a validation set (training:validation ratio was 1,559:803, roughly 2:1). The algorithm was optimized with parameters $\eta = 0$, $M = 20$. We use three measures of prediction quality: the negative logarithmic likelihoods (total, for incorrect and for correct answers):

$$-LL = -\frac{1}{2\log 2} \left(\frac{1}{|N|} \sum_{i \in N} x_i \log p_i + \frac{1}{|N|} \sum_{i \in N} (1 - x_i) \log(1 - p_i) \right) \tag{22}$$

$$-LL_+ = -\frac{1}{2\log 2} \left(\frac{1}{|N_+|} \sum_{i \in N_+} \log p_i \right) \tag{23}$$

$$-LL_- = -\frac{1}{2\log 2} \left(\frac{1}{|N_-|} \sum_{i \in N_-} \log(1 - p_i) \right) \tag{24}$$

$$MAE = \frac{1}{|N|} \sum_{i \in N} |x_i - p_i| \tag{25}$$

$$RMSE = \sqrt{\frac{1}{|N|} \sum_{i \in N} (x_i - p_i)^2}, \tag{26}$$

where the notation is: N is the set of question-user records, divided into the set N_+ of correct responses and the set N_- of incorrect responses. The correctness of each question is $x_i = 0, 1$ and p_i is the predicted probability of correct response. As defined, all measures would equal 0.5 if prediction were done by a coin toss, and 0 if prediction was perfect ($x_i = p_i$). As performance benchmarks, we take 1) "chance" prediction with p_i being the mean correctness in the training set (same value for all problems) and 2) "specific chance" – an improved version, where p_i is the problem-specific mean correctness in the training set (different for each problem). To average over fluctuations, we repeated random training:validation splitting 54 times, each time recording observed correctness and predicted probabilities for each user-problem observation.

We wanted to avoid measuring accuracy on events where user is submitting a problem having no prior exposure to the problem's LOs. Thus, we can restrict the validation set events to those with at least 1 prior exposure (Table 1).

Table 1. 13 LOs with more than 3 problems. Accuracy measures on interactions after 1 exposure to an LO (320,231 predictions)

	chance	specific chance	$M = 20$
$-LL$	0.462	0.355	0.474
$-LL_+$	0.335	0.250	0.393
$-LL_-$	0.715	0.564	0.636
MAE	0.457	0.340	0.389
RMSE	0.473	0.413	0.482

We see that LL_- is frustratingly high for all three methods of prediction, indicating a large presence of false positives. The benchmark chance algorithms also struggle there. If we predicted correctness by rounding the probability of correctness, we can compare the frequencies of false positives etc. directly (Tables 2), and the comparison is favorable to knowledge-tracing. Increasing the minimal number of exposures has the

Table 2. 13 LOs with more than 3 problems. Accuracy measures on interactions after 1 exposure to an LO (320,231 predictions)

knowledge tracing	observed $-$	observed $+$
predicted $-$	17.5%	21.6%
predicted $+$	15.9%	45.0%

special chance	observed $-$	observed $+$
predicted $-$	14.2%	8.95%
predicted $+$	19.2%	57.60%

tendency to reduce the number of false negatives, but typically at the cost of more false positives (higher values of LL_- show that false positives are a persistent problem for all three prediction methods. However, requiring more exposures restricts the data to fewer LOs, thus altering the game completely. As an extreme measure, if we require 7 exposures, we are left with only 1,433 observations, and specific chance predicts correct outcome for all of them, whereas the knowledge tracing prediction looks better (Table 3):

Table 3. 13 LOs with more than 3 problems. Accuracy measures on interactions after 7 exposures to an LO (1,433 predictions)

knowledge tracing	observed $-$	observed $+$
predicted $-$	12.6%	12.2%
predicted $+$	7.4%	67.8%

special chance	observed $-$	observed $+$
predicted $-$	0%	0%
predicted $+$	20%	80%

CONCLUSION

REFERENCES

1. William J Hawkins, Neil T Heffernan, and Ryan SJD Baker. 2014. Learning bayesian knowledge tracing parameters with a knowledge heuristic and empirical probabilities. In *International Conference on Intelligent Tutoring Systems*. Springer, 150–155.