International Journal of Artificial Intelligence in Education manuscript No.
(will be inserted by the editor)

Are We There Yet? Evaluating the Effectiveness of a Recurrent Neural Network-Based Stopping Algorithm for an Adaptive Assessment

Jeffrey Matayoshi · Eric Cosyn · Hasan Uzun

Received: date / Accepted: date

Abstract Many recent studies have looked at the viability of applying recurrent neural networks (RNNs) to educational data. In most cases, this is done by comparing their performance to existing models in the artificial intelligence in education (AIED) and educational data mining (EDM) fields. While there is increasing evidence that, in many situations, RNN models can improve on the performance of these existing methods, in this work we take a different approach. Rather than directly comparing RNNs with other models, we are instead interested in the results when RNNs are combined with one of these existing models. In particular, we attempt to improve the performance of ALEKS ("**A**ssessment and **LE**arning in **K**nowledge **S**paces"), an adaptive learning and assessment system based on Knowledge Space Theory, through the use of RNN models. Using data from more than 1.4 million ALEKS assessments, we first build an RNN classifier that attempts to predict the final result of each assessment. After verifying the accuracy of these predictions, we develop our stopping algorithm, with the goal of improving the efficiency of the ALEKS assessment by reducing the total number of questions that are asked. Based on this stopping algorithm, we give a comprehensive analysis of the possible effects it would have on students. We show that the combination

J. Matayoshi
McGraw Hill ALEKS
15460 Laguna Canyon Road, Irvine, CA 92618, USA
E-mail: jeffrey.matayoshi@aleks.com

E. Cosyn
McGraw Hill ALEKS
15460 Laguna Canyon Road, Irvine, CA 92618, USA
E-mail: eric.cosyn@aleks.com

H. Uzun
McGraw Hill ALEKS
15460 Laguna Canyon Road, Irvine, CA 92618, USA
E-mail: hasan.uzun@aleks.com

of an RNN with the ALEKS assessment can reduce the average assessment length by over 26%, while a high degree of accuracy is maintained.

Keywords Recurrent neural networks · Adaptive assessment · Knowledge Space Theory · Deep learning

1 Introduction

Over the last several years, deep learning techniques have achieved dramatic breakthroughs in many scientific fields [30,44]. Arguably, the first of these came in the area of computer vision [42], with subsequent advancements appearing in fields such as language modeling [18,43,56], speech recognition [32, 69], game playing [71,72], and machine translation [79]. Influenced in no small part by these achievements, models and applications using neural networks are moving into the education domain, with these concepts appearing in studies throughout the artificial intelligence in education (AIED) and educational data mining (EDM) fields. In particular, because of the sequential nature of many types of educational data, recurrent neural networks (RNNs) are being used more frequently in the educational literature [5,37,39,68,81].

In comparison, there are also many longstanding and well-studied models of learning and assessment that are used throughout the AIED field. These include models and theories such as Bayesian Knowledge Tracing (BKT) [2, 15,84], Knowledge Space Theory (KST) [23,25,26], and Performance Factors Analysis (PFA) [59]. These methods have been analyzed and vetted by a substantial body of research, and a large number of adaptive learning and assessment systems in use are based, at least in part, on these frameworks [8,11,25, 34,48,60] (see also [1,17], and the references therein, for further examples). Given the pervasiveness of these methods, a natural progression is to look at the viability of using deep learning to improve on them. As such, much of the recent work in the AIED field related to neural networks has focused on analyzing their effectiveness by comparing their performance to that of various existing methods and techniques.

For example, in the area of student modeling, [63] introduced an RNN-based model of student knowledge that was compared and contrasted with more traditional approaches, such as BKT. While the initial results seemed promising, follow-up studies [40,77,80] revealed a more nuanced picture. Specifically, [80] discussed issues with the data and methodologies in [63], with these issues complicating the comparisons that were made. Additionally, both [40] and [77] presented examples where the RNN model was matched, or even outperformed, by models based on BKT and item response theory (IRT). These results were synthesized in [78], which concluded that deep learning models, while showing promise in the education field, are by no means guaranteed to outperform other educational models. This conclusion is supported by subsequent work such as [49], which had BKT outperforming RNN models on one measure, yet simultaneously performing worse on others.

Several other examples exist of RNNs being applied to problems related to student modeling and prediction. In [46], RNNs were used to model student learning gains, and it was shown that the RNN models were more accurate and efficient in comparison to BKT models. Another relevant study is [36], where RNNs were applied to the task of making course grade predictions, based on the student's history of previous courses and grades. These predictions were then used as part of a novel course recommendation system that generated a suggested list of prerequisite courses for the student. Yet another example is contained in [39], where RNNs were used to classify students into groups based on their learning behaviors; the results indicated that the RNN models were at least as good as the methods traditionally used for this task. Finally, one additional study from this area is [52], where RNNs were used as part of a model of student knowledge retention that extended and improved on a basic forgetting curve.

RNNs have also been used for other topics, such as in recent studies of models for sensor-free affect detection [5,37]. In these works, RNN models were fed sequences of actions by students, and the models then used these actions to classify students into various affective states. In comparison to previously used techniques, [5] reported a significant overall improvement when applying RNNs to affect detection; furthermore, while the results in [37] were more mixed, they still showed a benefit to using RNNs.

While it is becoming clear that deep learning and RNNs have a place in the AIED and EDM fields, the discussion in the preceding paragraphs demonstrates that there are instances where the use of another model may be preferred. Motivated by these observations, the current work takes a slightly different approach. Rather than directly comparing and contrasting deep learning with other AIED models, we are instead interested in the results when deep learning is used *in combination* with one of these models. Specifically, ALEKS ("**A**ssessment and **LE**arning in **K**nowledge **S**paces") [54] is an adaptive learning and assessment system based on KST. At the core of the ALEKS system is an adaptive assessment that aims to precisely and efficiently identify the topics in an academic course that a student knows, as well as identify the topics she is ready to learn next. Our goal is to augment the KST-powered ALEKS assessment with the classification strengths of an RNN model.

The specific focus of our efforts in this study is ALEKS Placement, Preparation and Learning (ALEKS PPL), a specialized product that has been developed to offer recommendations for placing students in post-secondary mathematics courses. In [50], an initial version of a stopping algorithm, also based on an RNN classifier, was introduced to improve the efficiency of the ALEKS PPL assessment. In this current work, we build on these previous results by evaluating an updated version of the stopping algorithm. The changes to the algorithm include a larger set of features for the machine learning classifier, as well as the removal of certain restrictions that slightly hindered the performance of the previous version. However, in comparison to [50], our main contribution is a significantly more comprehensive and detailed evaluation of the stopping algorithm and its various effects. Thus, in addition to a more in-depth analysis of the performance of the predictive models, we spend a considerable amount of time attempting to understand the effects of the stopping algorithm on students if it were to be operationalized; for example, we look at the time that the algorithm can save students. Finally, we propose and evaluate an alternative version of the stopping algorithm that is more flexible and easier to maintain in a production environment.

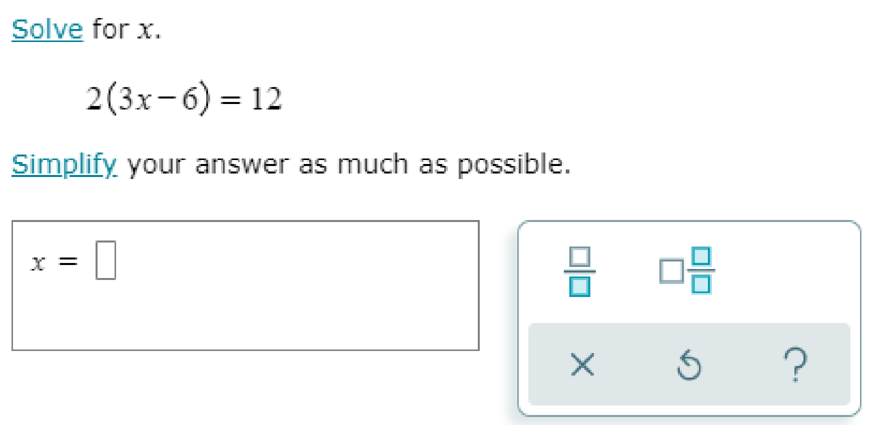
The outline of the paper is the following. We begin with background information, where we give a brief introduction to KST and the KST-based criteria currently used for stopping the ALEKS PPL assessment. We then describe our experiments and the RNN model building procedure. Based on the RNN model, we define an updated stopping algorithm that we apply to our test data, and we attempt to understand its effects with a comprehensive analysis of the results. Finally, we evaluate the aforementioned alternative version of the stopping algorithm, and we compare and contrast the performance of the two different versions of the algorithm.

2 KST, ALEKS, and the Current Stopping Rule

In this section we give a brief introduction to KST and the ALEKS assessment. For more a detailed introduction to these subjects, we refer the reader to [25, 26]. In KST and ALEKS, an *item* is a problem type that tests a discrete unit of the curriculum, with each item being composed of a collection of examples that are designed to be equal in difficulty. A *knowledge state* is a set of items that a student knows, that is, the student has the ability to solve. The *knowledge space* is the collection of all such possible knowledge states. In general, the number of knowledge states in the knowledge space is much less than the total number of possible sets of items. This is because the items share specific relationships that determine the collection of knowledge states. For example, a knowledge state that contains an advanced item will necessarily contain more elementary items that test concepts required for the mastery of that advanced item.

Figure 1 contains a screen capture of an example math item titled "Introduction to solving an equation with parentheses." This item tests a student's ability to apply the distributive property and solve a linear equation. As such, it is a prerequisite item for more advanced equation solving items. For example, if a student knows an item with a name such as "Solving a linear equation with several occurrences of the variable and distribution," it stands to reason that this same student is able to solve the problem in Figure 1, which covers prerequisite concepts needed for this more advanced item.

The goal of the ALEKS assessment is to discover the student's knowledge state within the knowledge space. The assessment employs an adaptive querying process, as the student's previous answers affect the choice of the next item to ask. Using this adaptive process, it attempts to identify the student's knowledge state with as few questions as possible. Among other things, the


Solve for x.

$$2(3x-6)=12$$

Simplify your answer as much as possible.

$x = \square$

Fig. 1 Screen capture of an ALEKS item titled "Introduction to solving an equation with parentheses." In addition to the problem statement, the screen capture shows the answer input box; tools for inputting fractions and mixed numbers; and buttons for clearing the answer box, undoing the last action, and obtaining help with the answer input tools.

assessment leverages the information contained in the knowledge space, such as the prerequisite relationships, to assist with this process.

Given a particular assessment question, the student's response falls into one of three categories:

– student inputs a correct answer;
– student inputs an incorrect answer;
– student clicks on the "I don't know" button.

Based on the student responses up to, and including, the current question, each item is assigned an estimated probability that it is contained in the student's knowledge state. The probabilistic nature of the assessment lends the flexibility that is needed to deal with careless errors and lucky guesses (see, for example, Chapter 13 in [26] for further information). Using the probability estimates, at all times the items under consideration are partitioned into the following categories by the ALEKS system:

– items that are most likely in the student's knowledge state (in-state);
– items that are most likely not in the student's knowledge state (out-of-state);
– the remaining items (uncertain).

The assessment terminates when the student's knowledge state is precisely identified; this happens when all of the items are classified as either in-state or out-of-state, with no remaining "uncertain" items. Alternatively, to prevent a student from having to answer too many questions, the assessment stops if a predefined maximum number of questions is reached. In both cases, the in-state items are returned as the best estimate of the student's knowledge state. Note that, because of the information contained in the knowledge space and the associated knowledge states, most items end by being classified as in-state or out-of-state without having been directly asked.

There are 314 items in ALEKS PPL, covering material from elementary mathematics to precalculus, and the ALEKS PPL assessment has a limit of 29 questions.[1] While the items used in ALEKS PPL also appear in other ALEKS products, they were specifically chosen for use in ALEKS PPL due to their relevance for placement in college mathematics. The *item score* of the student is simply the number of items that are in the student's knowledge state (i.e., the in-state items) at the end of the assessment. The *percentage score* is the ratio of the item score over 314 (the total number of items). From these scores, ALEKS PPL recommends placement in a course. The default recommendation is based on the cut scores in Table 1. Next to each of the cut scores we have listed the corresponding course(s) from the typical sequence of math courses in use at many U.S. colleges.

Table 1 ALEKS PPL placement recommendations

Placement category	**Item Score**	**Percent Score**	**Course placement**
1	< 44	$< 14\%$	Basic Math/Prealgebra
2	≥ 44	$\geq 14\%$	Beginning Algebra
3	≥ 94	$\geq 30\%$	Intermediate Algebra
4	≥ 144	$\geq 46\%$	College Algebra
5	≥ 192	$\geq 61\%$	Precalculus/Business Calculus
6	≥ 239	$\geq 76\%$	Calculus I

Approximately 97% of ALEKS PPL assessments reach the maximum limit of 29 questions and thus end with a number of uncertain items. There are several reasons for this. To start, most ALEKS items have an open-ended free response interface, and are thus prone to careless errors, or "slips," during an assessment (in comparison, "lucky guesses" are much less common). Such careless errors, and the resulting misrepresentation of what the student knows, affect the number of questions it takes to accurately identify the knowledge state (see [62], and the references therein, for an enlightening discussion of these issues in a related context). Further complicating the process of extracting reliable information from student responses is that, as discussed in [51], there is evidence that as an ALEKS assessment progresses, the behavior of some students may change. For example, students may experience a type of assessment fatigue, making them less likely to attempt to answer a question later in the assessment, and thereby decreasing the chance that uncertain items will be moved to the in-state category. Additionally, from a more theoretical standpoint, the combinatorial nature of knowledge spaces presents difficulties of its own when attempting to identify the knowledge state of a student. The

[1] Students actually answer up to 30 questions when accounting for a randomly chosen question asked during the assessment and used for validation and other statistics.

number of states in a knowledge space typically grows exponentially as a function of the number of items; since many knowledge spaces used in practice contain several hundred (or even over a thousand) items, counting the exact number of states quickly becomes an intractable problem. For example, in the specific case of the knowledge space used for ALEKS PPL, the number of states is estimated to be on the order of 10^{23}.

While the previous paragraph describes several reasons that explain why precisely identifying a student's knowledge state is not an easy problem, the adaptive and probabilistic nature of the ALEKS assessment allows it to deal appropriately with the challenges arising from both the open format of the items and the students' shifting behaviors [25,26]. As for the combinatorial difficulties, it must be noted that, if the assessment stops before removing all the uncertain items, there are (potentially many) remaining candidates for the actual knowledge state of the student. In particular, the in-state items can be viewed as a lower bound on the items the student knows, while the complement of the out-of-state items is then the corresponding upper bound. In practice, it turns out that the set of in-state items provides an accurate description of the student's standing by the time the assessment ends (see, for example, studies evaluating the validity and reliability of the ALEKS assessment in [16,22,25]).

There is, however, a particularity of ALEKS PPL that distinguishes it from other KST applications. The goal of the assessment here is not so much as identifying the detailed knowledge state of the student as simply gathering enough information for an accurate placement in one of six categories. Building on this idea, it is possible that the course placement recommendation for a student can be identified before the full 29 questions of the assessment are asked. Our goal is to take advantage of this fact and to develop a stopping algorithm for the ALEKS PPL assessment that in many cases can, with high accuracy, identify the appropriate course placement recommendation with fewer than 29 questions. To do this, we first build several machine learning classifiers that aim to predict the final course placement recommendation for a student, based on the (partial) information of the assessment up to a given question number. Once this is done, we can use these classifiers to implement an actual algorithm for stopping an ALEKS PPL assessment.

3 Experimental Setup

The data for our experiments consist of 1,449,625 ALEKS PPL assessments, with each assessment being taken by a unique student for placement purposes in a college or university setting (typically in their first year, but not always). For this analysis, we use only full-length assessments consisting of 29 questions (as mentioned previously, roughly 97% of ALEKS PPL assessments reach this maximum number of questions). The assessments were taken during a time period starting in March 2012 and ending in October 2018, and the distribution of the course placement recommendations is given in Figure 2. After being processed, the data used to train our models are sequential in nature, with

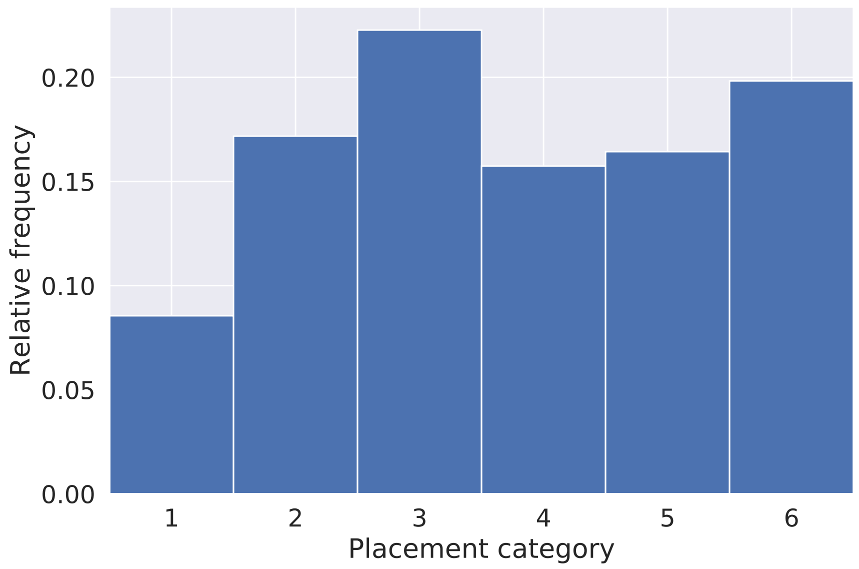


Fig. 2 Relative frequency histogram of course placement recommendations from the 1,449,625 full-length assessments in our data set.

each assessment generating one sequence; each sequence consists of 29 steps, with one step for each question on the assessment. Of the assessments in the data set, 50,000 are randomly chosen and used for a held-out test set, another 50,000 are randomly chosen and used for a validation set to tune hyperparameters and compare several models, and the remainder (1,349,625) are used for training our models. As is standard practice when applying deep learning models to large data sets [65,73], we use a single partition of our data into training, validation and test sets, rather than applying k-fold cross-validation or a similar method. This is in part due to the computationally expensive nature of training deep learning models, but it is also because of the important fact that, with such a large test set, we can expect the results to generalize to our full data set.

For our classification models, the target (ground truth) label for each sequence is determined by the course placement recommendation made by the ALEKS system using the full 29 questions from the assessment. Formally, using the knowledge state returned by the ALEKS system after question n (i.e., the set of in-state items after question n), the student's percentage score is computed to find the recommended course placement based on Table 1; we refer to this placement recommendation as C_n. In this context, our target labels then correspond to C_{29}. Thus, the results of the ALEKS PPL assessment can be viewed as a multiclass classification problem with six different class labels, one for each of the possible course placement recommendations. Under this framework, we are interested in building classifiers that can use only the first n questions of the assessment, with $n < 29$, to reliably predict the course recommendation returned by the full-length assessment. Note that the purpose of this work is *not* to validate the current course placement recommendations made by ALEKS PPL; validity of the ALEKS system is investigated in other works such as [16, 25, 66]. Instead, the goal of our study is to match the current recommendations made by the full-length ALEKS PPL assessments.

As an aside, in addition to the classification model described in the previous paragraph, during this research we also explored other models that estimated how much the student's percentage score would change during the remainder of the assessment (rather than attempting to predict the final course placement). This approach has some similarities with other stopping rules that have been applied to student modeling. For example, works such as [38, 41, 45, 67] applied stopping rules based on whether the predictions from a student model had stabilized. However, while this approach seemed promising initially, its performance ultimately proved to be inferior to that of the classification model that predicts the final course placement.

4 Models

For our recurrent neural network models, we use two different recurrent units: gated recurrent units (GRU) [12] and long short-term memory (LSTM) units [33]. We include both models in our experiments since there currently is not a consensus that one architecture or the other gives superior performance, as several studies and comparisons have not revealed a clear winner; these include examples both within the education domain [5, 37, 39, 68], as well as from the broader machine learning community [14, 83]. Both RNN models use a softmax output function that assigns an estimated probability to each of the class labels, and the cross-entropy loss is computed at each step in the sequence (i.e., after each question in the assessment) using these probabilities. Additionally, as a baseline comparison, we also build a set of logistic regression (LR) classifiers, one for each question number, where each logistic regression is trained only on the data up to that point in the assessment.[2] To handle the multiclass aspect of our problem, the logistic regression classifiers use a one-vs.-rest approach, where a separate model is trained for each class label (in comparison to a true multinomial logistic regression, the one-vs.-rest approach gave better results on our validation set). Figures 3 and 4 contain graphical representations of the models.

When building our models, we use two different methods for generating our features. Our first method views the classification problem based on the actions of the students during the assessment. Specifically, as our features we use the

[2] We also experimented with a similar methodology for the RNN models. That is, rather than building a single RNN model, we built separate models for each question number. However, since the performance was very similar to that of the single RNN model, and due to the extra complexities required by this method, we did not pursue this approach further.

items asked and the responses given during the assessment, and we refer to this set of features as the student-centered approach. For the student-centered approach, we have three variables per item, with each variable representing a possible outcome given a student response; recall that these responses can take the form of a correct answer, an incorrect answer, or an "I don't know" answer. This approach requires a total of $3 \times 314 = 942$ independent variables to represent all the possible combinations of responses and items. For our RNN models, each response is encoded in a different vector, with the n-th vector containing the response at question n, and only the response at question n. For example, if question 15 was answered incorrectly, the 15th vector contains a "1" in the column representing an incorrect answer to the item asked at question 15, and a "0" in each of the other 941 columns. Then, as we have 29 questions in each sequence, the entire set of features for an individual student is contained in an array of size 29×942.

Since the logistic regression models are unable to process sequential data, the features for the student-centered logistic regression are slightly different from those of the RNN models. That is, at question n the logistic regression features include the response from question n, along with all of the previous $n - 1$ responses; this means the feature vector has n nonzero components at question n. These features are then used to train the logistic regression model for question n; as mentioned previously, while the same RNN model is used at all points in the assessment, we train separate logistic regression models for each question number. Because of this, the logistic regression models are at a relative disadvantage in comparison to the RNN models, as each individual logistic regression is only trained on data up to that specific point in the assessment. Contrast this with the RNN models, which are fed the entire sequences of data when training; it seems plausible that this extra information is used by the RNN models to improve their predictive accuracy.

Our second method uses the actual item categorizations of the ALEKS assessment as features. Recall that at each point in the assessment an item is categorized as being either in-state, out-of-state, or uncertain. Thus, using this assessment-centered approach, we again require $3 \times 314 = 942$ independent variables, in this case to represent all possible combinations of assessment categories and items. The n-th vector contains the categorization of the items by the assessment after question n. Specifically, each in-state item has a value of "1" in its in-state column and a value of "0" in each of the out-of-state and uncertain columns; each out-of-state item has a value of "1" in its out-of-state column, and so on. Note that for this approach the RNN models and the logistic regression models all use exactly the same features.

As discussed previously, it was shown in [51] that the response patterns of students change throughout the course of the assessment; as one example, it was mentioned that students may be more reluctant to attempt a problem later in the assessment. Such evidence is what motivates the student-centered approach, which attempts to use the specific responses of the student to predict the class label they are likely to end the assessment with. In comparison, the ALEKS assessment takes the response of the student and, being adap-

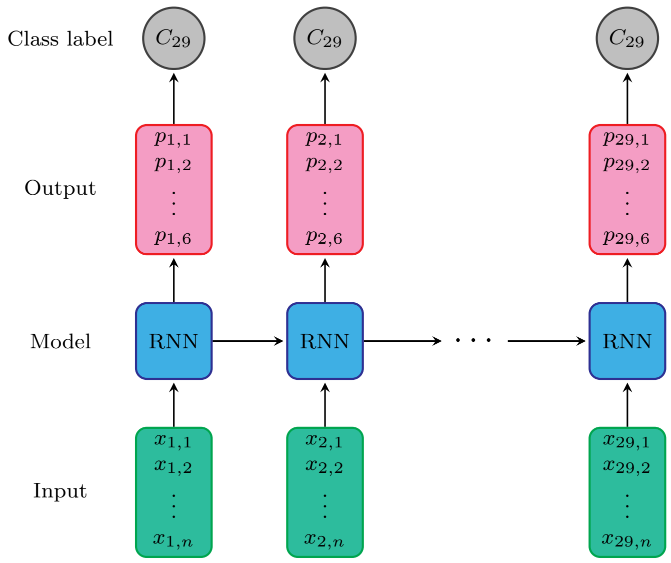

Class label
C_{29}
C_{29}
C_{29}
Output
$p_{1,1}$ $p_{1,2}$ $\vdots$ $p_{1,6}$
$p_{2,1}$ $p_{2,2}$ $\vdots$ $p_{2,6}$
$p_{29,1}$ $p_{29,2}$ $\vdots$ $p_{29,6}$
Model
RNN
RNN
$\cdots$
RNN
Input
$x_{1,1}$ $x_{1,2}$ $\vdots$ $x_{1,n}$
$x_{2,1}$ $x_{2,2}$ $\vdots$ $x_{2,n}$
$x_{29,1}$ $x_{29,2}$ $\vdots$ $x_{29,n}$

Fig. 3 Graphical depiction of the RNN model. The variable $x_{i,j}$ represents the j-th input feature at question i, where the total number of features, given by n, is either 942 or 1884. The value of $p_{i,j}$ in the output represents the predicted probability of class label j at question i, and C_{29} is the ALEKS PPL course placement recommendation at the end of the assessment (i.e., after question 29). While the RNN model generates unique predictions after each step in the sequence, the target label (C_{29}) used for computing the cross-entropy loss is the same at each of these steps.

tive in nature, updates its expectation of what items the student knows (or doesn't know) and categorizes the items accordingly. This information from the assessment system is lost when taking the student-centered approach, and thus the assessment-centered approach makes for an interesting comparison. Conveniently, it should be mentioned that each approach uses the same number of features (942), facilitating comparisons between the two. Additionally, as an extra set of features that was not included in [50], we also evaluate the performance when the assessment-centered and student-centered features are combined (for a total of 1884 variables). Using a larger number of features requires a correspondingly larger set of model parameters, exposing the model to overfitting issues. However, given the large number of assessments in our training data, we expect that any possible overfitting will be more than compensated for by the added information from the combined set of features, thus yielding the best performing model. As we will see shortly, this assumption is correct, and leads to a slight improvement over the classifiers from [50].

5 Model Evaluation

All of the models are repeatedly trained on the 1,349,625 assessments in the training set, while the results on the 50,000 assessments in the validation set

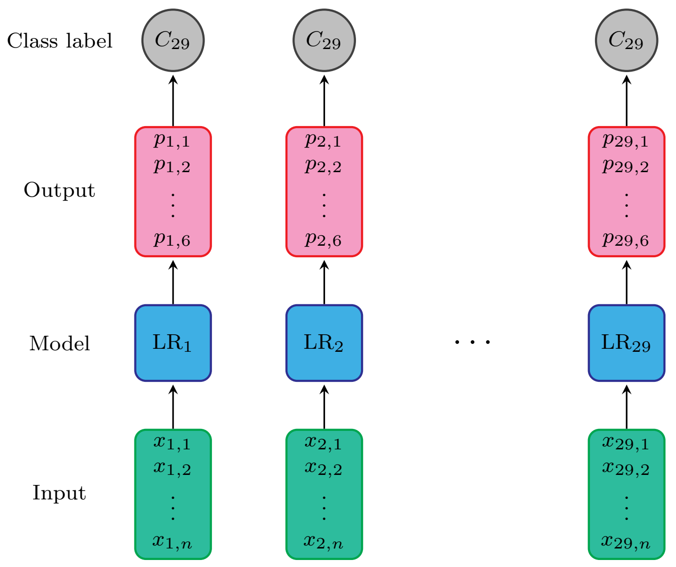


Fig. 4 Graphical depiction of the logistic regression model. The variable $x_{i,j}$ represents the j-th input feature at question i, where the total number of features, given by n, is either 942 or 1884. The value of $p_{i,j}$ in the output represents the predicted probability of class label j at question i, and C_{29} is the ALEKS PPL course placement recommendation at the end of the assessment (i.e., after question 29). In contrast to the RNN model, 29 separate logistic regression classifiers are trained, one for each question number in the assessment.

are used to tune the various hyperparameters (with the held-out test set then being used for the evaluation and analysis of our final models, beginning in the next section). For each of the neural network architectures (i.e., LSTM or GRU), the number of hidden layers, the sizes of the hidden layers, and the learning rate are tuned on the validation set. We also experiment with the usage of batch normalization [35], a method for normalizing the inputs of the neural network layers that, in many cases, improves the performance of the model [30,35,70]. To help prevent overfitting, in all of our RNN models we apply early stopping [65] and dropout [28,73]; these regularization techniques are commonly applied to neural networks, as they help improve the ability of the models to generalize to new data [30]. For the logistic regression models, the only tuned hyperparameter is the strength of the L_2 regularization; most likely due to the large amount of data in the training set, lower regularization improves the performance, with the best performing logistic regression model essentially removing the regularization.

The logistic regression models are trained using the scikit-learn [61] Python library. As this research took place over a period of several years, we initially built and trained our RNN models using a combination of Keras [13] and Theano [74]; however, we switched to PyTorch [58] for the latter portion of this work. To test out different combinations of hyperparameters for the RNN models, we use an iterative approach, in which the results from previous training runs are used to inform the choices of hyperparameters for subsequent runs. Due to the relatively long training time, as well as the large number of hyperparameters, we use this manual process for tuning the hyperparameters instead of more traditional techniques, such as grid search. On the validation set, we experiment with models using anywhere from one to six hidden layers, and we vary the sizes of the hidden layers from as little as a few hundred units, up to a few thousand. For our hidden unit activation functions we exclusively use rectified linear units (ReLU), and for all our models (both logistic regression and RNN) we use cross-entropy loss.

For each classifier architecture, the best results from the performance on the validation set are shown in Table 2, where the accuracy, Matthews correlation coefficient, and log loss are reported for two different points in the assessment; we show the results after question 15, which gives an idea of the performance roughly halfway through the assessment, and after question 25, which highlights the performance near the end of the assessment. The Matthews correlation coefficient, introduced in [53] and extended to the multiclass case in [31], is a statistic for measuring the quality of a classifier. It performs well with unbalanced data [6] and has been suggested as being more informative in comparison to other measures such as the accuracy and F1 score [9,64]. To apply the Matthews correlation coefficient, we use the class with the highest probability estimate as the predicted class label, which is then compared to the true class label (in the case of binary class labels, Matthews correlation coefficient is actually equivalent to the phi coefficient). Also, as a point of comparison for the classifiers, the "ALEKS" row reports the accuracy and Matthews correlation coefficient if we were to use the student's current course placement (i.e., C_n) as defined by the in-state items at that time (after either question 15 or 25).

While all of the classifier models perform relatively well, the RNN models show small, but consistent, gains over the logistic regression models. Additionally, while the assessment-centered model performs better than the student-centered model in all cases, the best results are from the combined model. Also of note is that the LSTM models seem to perform slightly better with both the student-centered and assessment-centered features; however, the strongest overall performance comes from the combined features using GRU hidden units. In the latter case, the GRU models might be less prone to overfitting due to their having fewer parameters than the LSTM models. Finally, the performance of the ALEKS assessment, while comparable to the other models at question 25, is notably worse at question 15.

The best performing RNN models consist of either two or three hidden layers. While we trained models with more hidden layers, the added depth did not improve the results, suggesting that a very "deep" representation of the data is not necessary to get optimal performance. Note that this is seemingly consistent, albeit in a slightly different context, with results from several studies [21,40,77,78] giving evidence of the relative lack of depth that is necessary to build accurate student models. The combined features LSTM model in Table 2 has an initial hidden layer with 1900 units and one additional hidden

Table 2 Comparison of different classifier models and architectures. Results shown are from the best performing models, as measured by the predictions applied to the validation set.

Model	**After question 15**			**After question 25**		
	Accuracy	**Matthews**	**Log loss**	**Accuracy**	**Matthews**	**Log loss**
ALEKS	0.648	0.579	—	0.921	0.905	—
Student-centered features:						
LR	0.820	0.781	0.409	0.921	0.904	0.202
GRU	0.824	0.786	0.394	0.925	0.909	0.178
LSTM	0.827	0.790	0.388	0.931	0.917	0.165
Assessment-centered features:						
LR	0.820	0.781	0.404	0.937	0.923	0.159
GRU	0.827	0.790	0.388	0.942	0.930	0.142
LSTM	0.828	0.791	0.387	0.943	0.931	0.140
Combined features:						
LR	0.825	0.788	0.391	0.939	0.926	0.154
GRU	0.830	0.793	0.379	0.944	0.932	0.136
LSTM	0.830	0.793	0.381	0.943	0.931	0.139

layer with 1200 units, while the GRU model has an initial hidden layer with 1900 units and two additional hidden layers, each with 1200 units.

As the best results come from the models using the combined set of features, we attempt to get a more precise measure of the differences in performance between these models by using McNemar's test [24,55]. In particular, we use McNemar's test to compare the accuracy values of the classifiers, as this procedure is recommended when a single partition of the data is used for training and evaluation (as opposed to a procedure that uses multiple partitions, such as cross-validation or resampling) [20]. McNemar's test evaluates the difference in accuracy scores by looking at the examples for which one model makes a correct prediction and the other makes an incorrect prediction; the resulting test statistic follows a χ^2-distribution with a single degree of freedom. For this analysis, we compare each of the three classifiers to each other after both questions 15 and 25, for a total of six comparisons. Due to these multiple comparisons, we also apply the Benjamini-Yekutieli procedure [4], with a threshold of 0.05, to control for false positives. Since each of our models are trained and evaluated on the same data, we use the Benjamini-Yekutieli procedure, instead of the more common Benjamini-Hochberg procedure [3], because the former is more conservative and can be applied regardless of the type of dependency that exists between the test statistics [4].[3] The results are shown in Table 3, where we can see that the differences in accuracy scores between the RNN and logistic regressions models are always statistically significant;

[3] It's worth mentioning that the statistical significance results are unchanged if the Benjamini-Hochberg procedure is applied instead.

Table 3 Results from applying McNemar's test to the predictions using the combined set of features. The $+/-$ columns show the number of examples from the validation set where the prediction from the first classifier is correct and the prediction from the second classifier is incorrect. Similarly, the $-/+$ columns show the number of times the first classifier is incorrect and the second is correct. An asterisk (*) next to the p-value denotes a statistically significant difference after applying the Benjamini-Yekutieli procedure with a threshold of 0.05.

Models	**After question 15**				**After question 25**			
	$+/-$	$-/+$	χ^2	p**-value**	$+/-$	$-/+$	χ^2	p**-value**
LR, GRU	807	1026	25.93	$\ll$0.001*	499	763	54.81	$\ll$0.001*
LR, LSTM	969	1182	20.89	$\ll$0.001*	624	833	29.69	$\ll$0.001*
GRU, LSTM	633	627	0.02	0.888	397	342	3.95	0.047

however, the differences in the accuracy scores between the RNN models are not significant.

6 Stopping Algorithm

Now that we have confirmed the accuracy of our classifiers, we can use the predictions from these classifiers to implement a stopping algorithm for the ALEKS assessment. The idea of the stopping algorithm, the full details of which are given in Algorithm 1, is the following. We first identify potential points at which to stop the assessment based on the confidence of the classifier. That is, our first criterion is that the most confident predicted class label is above a certain threshold, α. Then, to ensure that our classifier has at least a minimal amount of data to work with, we also require that each assessment has asked at least five questions before the stopping algorithm is activated.

In comparison to the algorithm outlined in [50], a couple of changes have been made that, based on further experiments on our validation set, led to an improvement in the performance of the stopping algorithm. First, we are not requiring that C_n (i.e., the course placement recommendation after question n, as determined by the student's current percentage score) is equal to the classifier's predicted class label. As this requirement can be viewed as an ensemble of the predictions from the classifier and the assessment, we initially believed it would lead to better performance. However, in some cases the classifier would return a very high predicted probability for the final course placement recommendation, but the previous version of the stopping algorithm would then wait for the ALEKS assessment's recommendation to match this prediction; thus, removing this restriction allows these assessments to end earlier. The second difference from [50] is that we are activating the stopping algorithm after 5 questions, rather than 10. As before, this restriction on the previous version was causing some assessments to continue longer than necessary, as in many cases the classifier's estimated probability was already very high before reaching this minimum number of questions.

Algorithm 1 Assessment stopping algorithm

```
Inputs:
$\alpha$, stopping threshold probability
$\mathbf{x}_n$, the input features of the classification model after question $n$
$P(k \mid \mathbf{x}_n)$, predicted probability of class $k$, $k = 1, \ldots, 6$, after question $n$
$K_n = \arg\max_{k=1,\ldots,6} P(k \mid \mathbf{x}_n)$; i.e., the most likely class after question $n$
$C_{29}$, the recommended course placement after question 29 (based on computing the student's percentage score and applying the cut scores in Table 1)

Iterations:
for $n = 5$ to 29 do
    if $n == 29$ then
        Return $C_{29}$
    else if $P(K_n \mid \mathbf{x}_n) > \alpha$ then
        Stop the assessment and return $K_n$
    end if
end for

Output:
The predicted course placement recommendation
```

Once we have defined our stopping algorithm, we can next evaluate the effect of the algorithm on the assessment's performance using our held-out set of test data. As measured by the values in Table 2, we saw in the previous section that while the assessment-centered features gave consistent gains over the student-centered features, the overall best performance came from the combined features; thus, for this evaluation we exclusively use the combined set of 1884 features. The effectiveness of Algorithm 1 will be evaluated on both the average assessment length and the accuracy of the predicted course placement recommendation as returned by the algorithm. Note that since, by definition, the placement recommendation returned after question 29 is the ground truth label, perfect accuracy would be obtained if no assessments are stopped before question 29.

The first set of results from the held-out test data are contained in Figure 5, where we plot the accuracy of the predicted course recommendation versus the average assessment length, for various probability thresholds (i.e., various values of α). The first thing to note is that the GRU model seems to give a small, but consistent, improvement over the LSTM model. In comparison, the performance of the logistic regression model, while strong, is clearly behind both of the RNN models. For example, at an accuracy of 0.99, the GRU model has an average assessment length of about 20.2 questions, while the logistic regression model has an average length of about 21.1 questions; at an accuracy of 0.995, the average lengths are 21.3 and and 22.7 questions, respectively. Additionally, at any accuracy rate of 0.995 or higher, the GRU model is a minimum of 1.4 questions better than the logistic regression, with the maximum difference being about 2.5 questions. It should be emphasized that since the course placement recommendation can have a significant effect on a student's subsequent classroom experience, maintaining the accuracy of

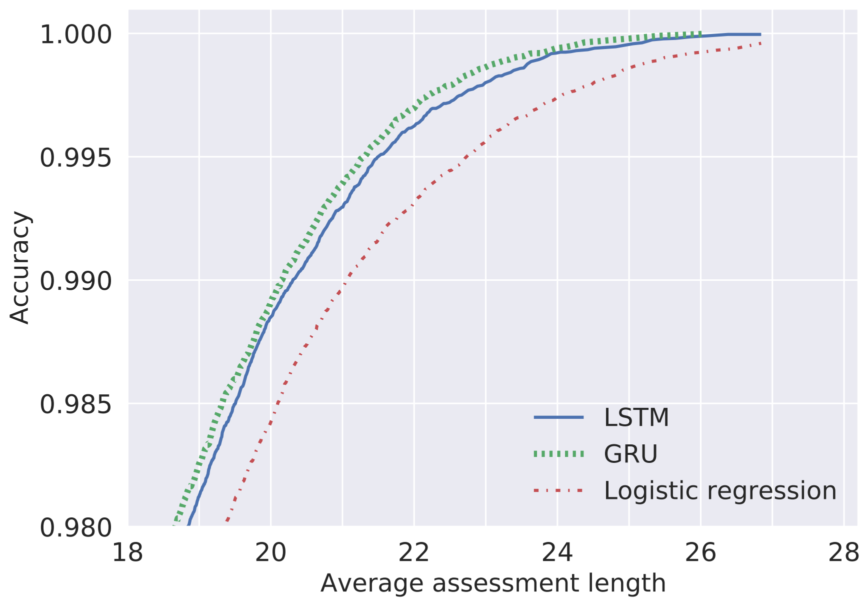


Fig. 5 Accuracy vs. average assessment length on held-out test data for the combined features models listed in Table 2.

the predicted recommendation is extremely important. Thus, in evaluating the performance of an actual implementation of the stopping algorithm, accuracy values above 0.995 are the most relevant, and these are the values for which the RNN models show the greatest gains over the logistic regression model.

Similar to our analysis in the previous section, we can compare the results of our stopping algorithm to the current performance of the ALEKS assessment by using the course placement recommendation after question n (i.e., C_n) as our prediction of the ground truth label. For example, if we stop every assessment on the held-out test set after question 28, using C_{28} as the predicted label results in an accuracy of 0.9795. Figure 6 shows the current placement accuracy after questions 5 to 29 where, for comparison, we have also included the results for the GRU model. We can see that there is a wide gap in performance between the two models. For example, after 20 questions we can see on the plot that ALEKS PPL has an accuracy of about 0.8; in comparison, when the average assessment length of the GRU model is 20, the corresponding accuracy is roughly 0.99. We thus observe a very large boost in accuracy and efficiency when the information from the current state of the ALEKS assessment is fed to the RNN classifier. Furthermore, the accuracy values for the ALEKS placement alone when it is stopped early show that, in many cases, the questions being asked at the end of the assessment are relevant to the student's final course placement. These results emphasize the

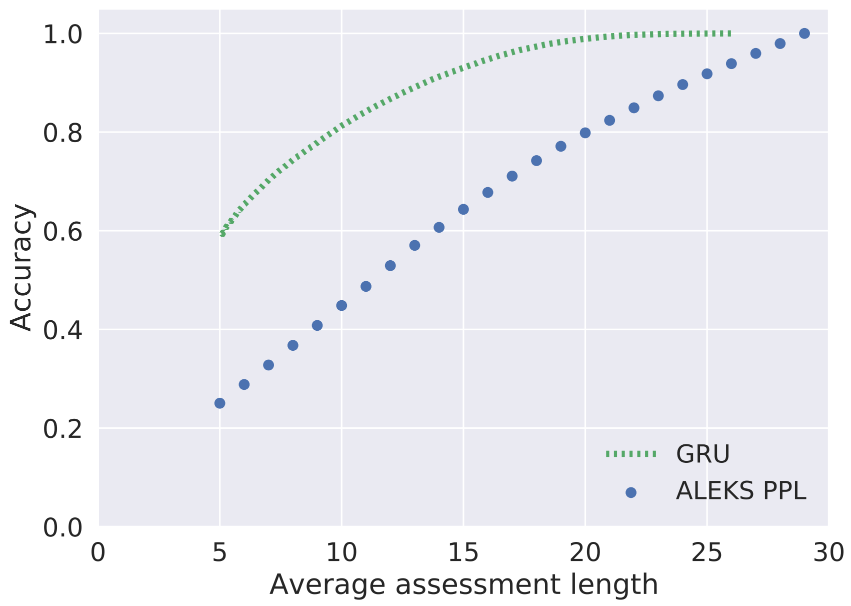


Fig. 6 Accuracy of the ALEKS PPL assessment when stopped after specific question numbers. (While the x-value for each ALEKS PPL data point is technically an "average," all the assessments represented by this data point have the exact same length.) For comparison, the performance of the GRU model with the combined set of features, using various threshold values α, is also shown.

importance of our classifiers, which are able to effectively separate the assessments that are still asking relevant questions from the ones that are no longer gaining useful information.

We next look at the performance of the classifiers using a specific value of α. Figure 7 contains a histogram of the assessment lengths after the stopping algorithm is applied, using both the GRU and logistic regression models with $\alpha = 0.99$. We can see that the majority of the assessments stop early, with only about 16.5% of the GRU assessments continuing for the full 29 questions; for the logistic regression model, roughly 21.7% continue for the full 29 questions. For these same models and α, Table 4 shows the results partitioned by the actual (ground truth) classification label. The best results, in terms of both assessment length and accuracy, are for the extreme labels 1 and 6, with the average assessment length showing a substantial reduction in both cases. In comparison, while still being acceptable, the gains are not nearly as large for labels 4 and 5. It is worth mentioning that these results closely parallel what was found in [22], where an evaluation of the reliability of the ALEKS PPL assessment was performed. In this evaluation, it was shown that ALEKS PPL has the least variability (or, equivalently, gives the most reliable results) for

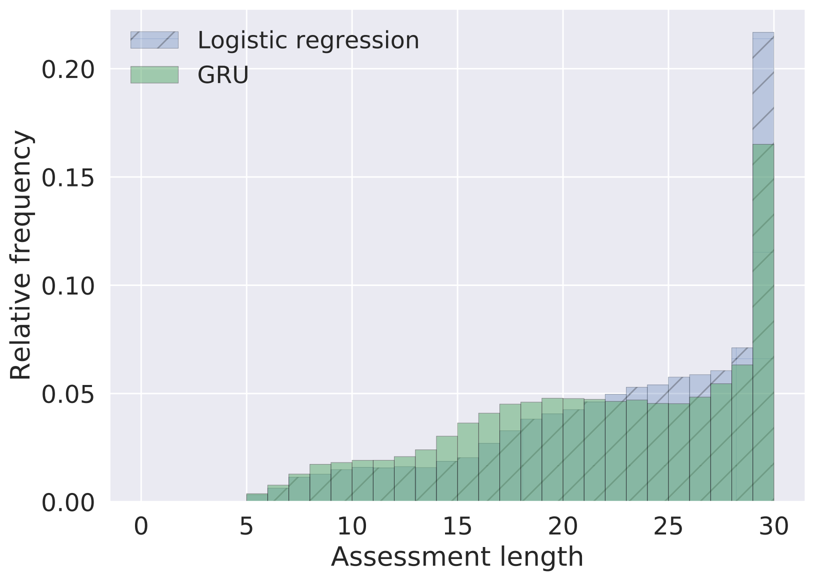


Fig. 7 Relative frequency histogram of assessment lengths after the stopping algorithm is applied on the held-out test data, using the combined features, the GRU and logistic regression models, and a threshold of $\alpha = 0.99$.

labels 1 and 6, while having the greatest variability for labels 4 and 5. Thus, it seems likely that this increased variability is a major reason for the weaker performance of the stopping algorithm with labels 4 and 5.

Table 4 Stopping statistics by ground truth label for the GRU and logistic regression models on held-out test data, using a threshold of $\alpha = 0.99$.

Class label	**Sample size**	**GRU**		**Logistic regression**	
		Average length	**Accuracy**	**Average length**	**Accuracy**
1	4357	17.48	0.9959	18.6	0.9970
2	8680	21.21	0.9944	22.23	0.9968
3	11108	21.89	0.9950	23.40	0.9947
4	7640	24.47	0.9908	25.71	0.9904
5	8259	25.34	0.9923	26.41	0.9932
6	9956	15.68	0.9971	17.35	0.9945
Total	50000	21.11	0.9943	22.42	0.9943

To further understand the results, we next look in more detail at the probability estimates returned by the GRU classifier after question 20. Figure 8 shows histograms of the final item scores (i.e., the number of items classified as in-state by ALEKS PPL at the end of the assessment), conditioned on the model classification probabilities above certain thresholds; note that we include the results for all the probabilities above a given threshold, regardless of whether or not the predicted class label is correct. For example, in the topmost plot we show the histogram computed from all the assessments in our test set for which the GRU classifier has a maximum probability greater than 0.8 at question 20; the other plots then show the histograms based on higher probability thresholds. The vertical (red) lines represent the different item cut scores given in Table 1, and from the figure we can see that the predictions with the highest probabilities occur away from these cut scores; in other words, by the time we reach the highest probability threshold of 0.99, hardly any item scores remain near the red lines. This makes intuitive sense; if an item score ends up being very close to a cut score, it is difficult for the classifier to know with much certainty on which side of the cut score the final item score will fall.

Furthermore, from Figure 8 we can also see that the classifier assigns the lowest probabilities to the predictions coming from the item scores corresponding to course labels 4 and 5 (i.e., the item scores from 144 to 238). In particular, raising the threshold to 0.99 results in a dramatic reduction of the number of probabilities for labels 4 and 5 (well in excess of the relative sample sizes of these categories in comparison to the other categories). Note that this corresponds to the weaker results shown in Table 4 for these labels. As before, the greater variability for labels 4 and 5 discussed in [22] seems to play a role here.

Next, we note that using a threshold of $\alpha = 0.99$, there are 285 assessments that are classified incorrectly by the GRU model. To get a better understanding of these misclassifications, Figure 9 shows a histogram of the difference between each item score and its closest cut score from Table 1. We can see that the majority of the data points tend to cluster around the cut scores; that is, 149 of these misclassified assessments (about 52%) are within two items of a cut score, and 222 assessments (about 78%) are within five items of a cut score. With that in mind, it should be noted that, if an item score is within a handful of items of a cut score, it is not clear which side of the cut score a student truly should be placed. Further information would be needed to determine which course placement is a better fit for the student.[4]

7 Assessment Duration

Building on the analyses in the previous section, we now attempt to understand the effect of the stopping algorithm on one important aspect of the assessment: the actual time required by each student to finish. Of the 50,000 assessments in our test set, we have access to the specific time spent on each

[4] To help with issues such as this, the PPL product has an option for remediating students who are not satisfied with their initial course placement. Thus, if a student falls short of a desired cut score, they can take advantage of this option and retake the PPL assessment, possibly placing into a higher course.

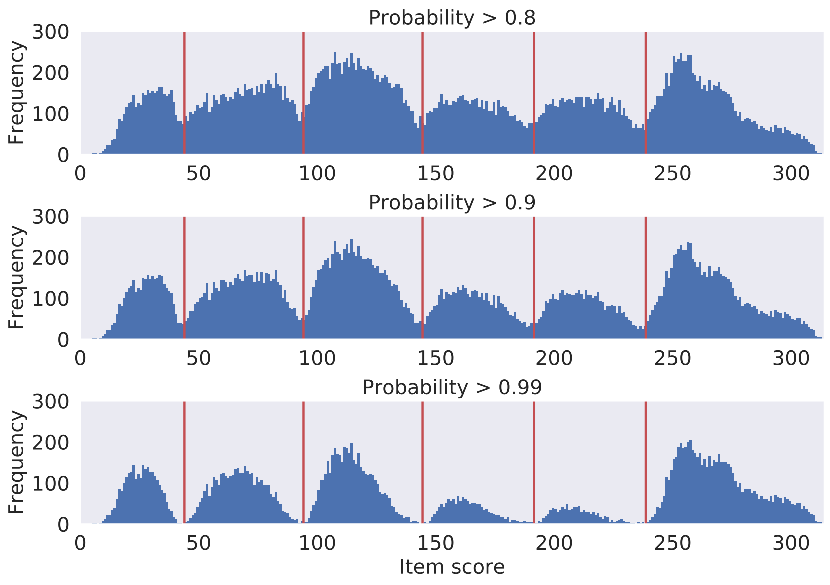


Fig. 8 Item score histograms for the final state using all 29 questions, conditioned on the GRU model classification probability at question 20. Vertical (red) lines correspond to the cut-scores from Table 1.

question for 45,470 assessments; due to technical reasons (mainly consisting of missing or corrupted data), we are unable to recover these data for the remaining 4,530 students. To begin, the striped (blue) bars in Figure 10 show the time distribution, in minutes, of these 45,470 assessments using all 29 questions. The mean and median durations are 93.6 minutes and 82.5 minutes, respectively. We can see from the figure that the distribution has a long tail, with the maximum duration being about 442 minutes. The solid (green) bars in Figure 10 then show the distribution of the time durations after the stopping rule is applied, using the combined features GRU model with a value of $\alpha = 0.99$. Here, we can see that the distribution has been shifted to the left, with the mean and median durations now being 70.1 minutes and 59.7 minutes, respectively. Thus, after applying the stopping algorithm, over half of the students are able to finish the assessment in less than an hour, with the average assessment duration being reduced by about 23.5 minutes.

Given that the distribution of the assessment times has many students with very long durations, we next look at the effect of the stopping algorithm on these extreme assessments. There are 3483 students who took longer than three hours to complete the assessment; these students have a mean duration of approximately 221.1 minutes. After applying the stopping algorithm, the mean duration for these students drops to 149.8 minutes (with a mean of 19.4 questions), a decrease of over 71 minutes. Furthermore, the number of

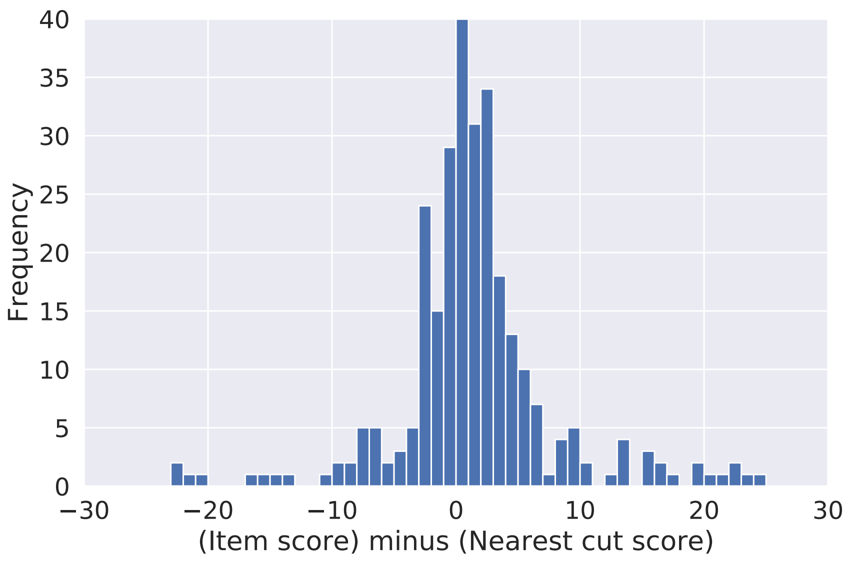


Fig. 9 Histogram of the difference between an item score and its nearest cut score from Table 1, for the 285 assessments that are classified incorrectly using the GRU model and a threshold of $\alpha = 0.99$.

assessments longer than three hours drops to 1299 (a reduction by a factor of ≈ 2.7). Thus, we see that the stopping algorithm has an even more pronounced effect on the students in the (right) tail of the distribution.

What is interesting about these extreme students is that the course placement distribution is very different from the overall distribution shown in Figure 2. As shown in Figure 11, the distribution is skewed to the right, with roughly 54% of the students placing into the highest category and another 24% placing into the second-highest category. At the moment, we do not have a good understanding of the reasons for this shift in the distribution, and our ability to analyze the situation is complicated by limited access to data on the subsequent course outcomes of these particular students. One view is that these students are very methodical and thorough while taking their assessments. That is, these are (possibly) students who are very motivated to place into a high course, and because of this they take their time, double-check their work, and only move on to the next problem when they are confident that they have solved the question correctly. Note that this scenario has some similarities to the findings in [57], where students were divided into clusters based on their learning profiles in the ALEKS system; while the results focused mostly on the learning aspect of ALEKS, as opposed to the assessment activity, it is worth noting that the cluster of students who worked at the slowest pace, yet simultaneously put in the most effort, ended with the highest scores. Another

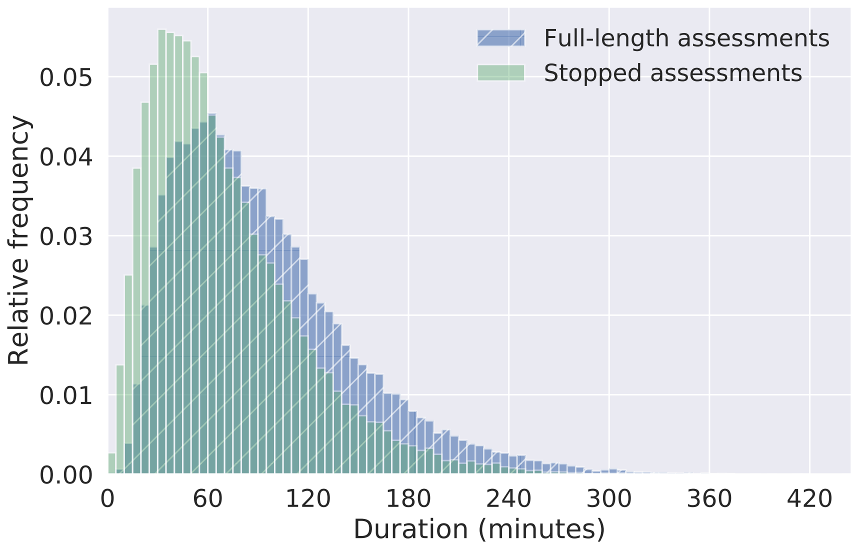


Fig. 10 Relative frequency histogram of assessment durations on the held-out test data. Solid (green) bars represent the assessments with the stopping algorithm applied, using the combined features GRU model and a threshold of $\alpha = 0.99$. The striped (blue) bars represent the durations from the full-length assessments using all 29 questions.

contributing factor to these long assessments is the fact that the highest performing students are likely to encounter the most difficult material on the assessment. This material can include topics such as trigonometry, and problems from these topics can take substantially longer to work through in comparison to the more elementary concepts.

A possible criticism of the above reasoning is that it doesn't completely fit with results from other studies specifically analyzing the response times of students. For example, the students in [29] had significantly longer response times on incorrect answers, in comparison to their response times on correct answers, while the students in [76] who took longer to answer questions were, on average, less successful; in other words, contrary to what we observe in ALEKS PPL, in both of these studies a longer response time was associated with weaker performance. One factor complicating this analysis is that the decision of whether or not to proctor the ALEKS PPL assessment is made by each individual institution, and it is possible that many of these assessments are being taken by students at home without any supervision. Thus, in such cases it would be feasible for some students to treat this as an "open book" exam, thereby using outside resources, such as textbooks or internet searches, to help find solutions to the assessment questions; the longer assessment times could then reflect the extra effort the students spend searching for help on

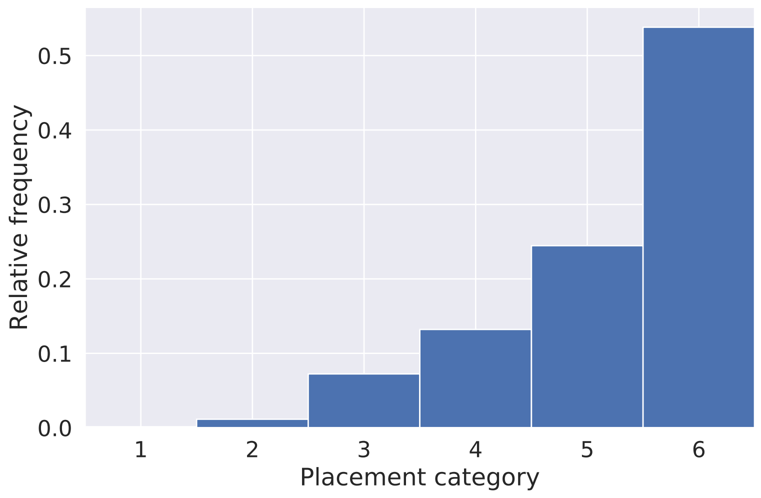


Fig. 11 Relative frequency histogram of course placement recommendations for the 3483 students in the held-out test data with full-length assessments longer than 180 minutes.

the assessment questions. Understanding more about the behavior of these extreme students is currently the subject of ongoing research.

8 Effect on the Final State

The goal of our next analysis is to evaluate the effect of the stopping algorithm on the knowledge state that is returned by the assessment. To that end, we focus on the differences between (a) the state that is returned when the assessments are stopped early and (b) the state from the full-length assessments. This is an important analysis, as the ALEKS PPL product has an option for remediating students who are not satisfied with their initial course placement, with the goal for them being to retake the ALEKS PPL assessment after the remediation and place into a higher course. In such cases, the knowledge state serves as the starting point for the student's remediation, and so the effect on this knowledge state should be measured and quantified. To do this, we use two different measures that are meant to quantify the amount of similarity between two states K and L. First, we compute the size of the *symmetric difference* of the two states, which is a measure that is commonly used within KST to compare different states [26]. The symmetric difference is simply the union of the two set differences $K \setminus L$ and $L \setminus K$. Second, we compute the *Jaccard distance*, which normalizes the size of the symmetric difference by the

Table 5 Statistics grouped by ground truth label for the size of the symmetric difference and the Jaccard distance between the state returned by the stopping algorithm and the state from the full-length assessment. For this analysis, we use the combined features GRU model on the held-out test data with a threshold of $\alpha = 0.99$.

Class label	**Symmetric difference size**			**Jaccard distance**		
	Mean	**Median**	**SD**	**Mean**	**Median**	**SD**
1	7.4	6	6.5	0.28	0.22	0.26
2	7.6	5	8.1	0.11	0.08	0.11
3	7.1	5	7.9	0.06	0.04	0.06
4	5.6	4	7.4	0.03	0.02	0.04
5	4.8	2	7.0	0.02	0.01	0.03
6	21.1	17	17.6	0.08	0.06	0.06

size of the union of the states:

$$\frac{|K \setminus L| + |L \setminus K|}{|K \cup L|}.$$

(In the above formula, the symbol $|\cdot|$ represents the size, or cardinality, of the given set.) The Jaccard distance computation returns a value from zero to one, inclusive, where zero indicates the sets are equal, and one indicates the sets are completely disjoint.

For our analysis, we use the results from the combined features GRU model on the test set, with a threshold of $\alpha = 0.99$. The results are shown in Table 5, where we can see some interesting contrasts between the two measures. For example, while the symmetric difference values for label 1 are roughly equivalent to most of the other labels, the Jaccard distance values are much higher; in the latter case, this is because the Jaccard distance is computed relative to the size of the state, and the label 1 states contain at most 43 items. Then, note that the symmetric difference values for label 6 are high, but they are relatively small after being normalized with the Jaccard distance. Additionally, students in this class are placed into the highest possible course (Calculus) and do not need to use the remediation mode of ALEKS PPL; thus, identifying their exact knowledge state is less critical.

Next, note that both the symmetric difference and Jaccard distance values are lowest for classes 4 and 5. Recalling the results from the previous sections, in which it was shown that the RNN classifier, as well as the overall stopping algorithm, has the weakest performance for these class labels, we can assume this is due to the fact that these assessments are not stopped as early as the rest. Specifically, in Table 4 we see that the average assessment lengths for labels 4 and 5 are roughly 24 and 25 questions, respectively, while the remaining labels range from about 16 questions to 22 questions. Since the stopping algorithm is less confident with assessments in categories 4 and 5, it allows them to run longer, and in such a case the state returned by the stopping algorithm is more similar to the state from the full-length assessment.

9 Training with Short-Term Course Placement Recommendations

The results from the previous sections show that the proposed stopping algorithm is both accurate and efficient, and that it can have a large positive benefit on the ALEKS PPL assessment. However, the procedure used to develop the stopping algorithm suffers from at least one drawback; namely, it requires access to the course placement recommendation of the assessment using all 29 questions, as this recommendation is used as the target (ground truth) label while training the RNN models. The problem with this approach is that if such a stopping algorithm were to be deployed to actual students, the majority of these assessments would be stopped before 29 questions, and we would no longer have access to the ground truth labels. According to Figure 7, upwards of 80% of the assessments are stopped early and would thus be missing labels; this presents obstacles for continually evaluating the performance of the stopping algorithm, or for further retraining of the RNN model.

One possible solution is to select a subset of assessments for which the stopping algorithm is not used. From a purely technical standpoint, this is a desirable procedure as it gives access to a set of labeled data that can be used to evaluate and retrain the RNN classifier. In comparison, from the student perspective this is suboptimal, as many students would lose the benefits of the stopping algorithm; in this scenario the issues of fairness and equality are significant and must be taken into consideration. To avoid this problem, an alternative way to evaluate the performance of the stopping algorithm is to directly study the outcomes of the course recommendations. That is, we can measure the validity of the ALEKS PPL course placement recommendations by looking at the performance of the students who enroll in the suggested courses. This is currently employed as part of our standard procedure for evaluating ALEKS PPL, and this practice could be applied to evaluate the performance of the stopping algorithm. However, this approach has its own drawbacks. For example, while looking at something like the pass/fail rates of students can indicate if there is a systematic problem with the ALEKS PPL recommendations, this information is not specific enough to use as the training labels for individual assessments, as these rates can vary widely based on factors such as the institution, course, or instructor. Thus, with this procedure the ground truth labels are still missing.

In the remainder of this section we explore an alternative model that attempts to address the issues discussed in the previous paragraphs. To obtain this new model, the only change we make to our previous model is how we select the target labels for the RNN. Rather than using the course placement recommendations from all 29 questions as the target labels, we instead try to predict the course placement on a shorter time scale. Specifically, at question n we use the course placement recommendation given by the ALEKS system at question $n + k$, where $k \geq 1$, as our target label. This means that each assessment now generates a sequence of $N - k$ steps, where N is the length of the assessment, as the remaining k questions no longer have target labels; note that the features for these $N - k$ steps are unchanged from before. Further-

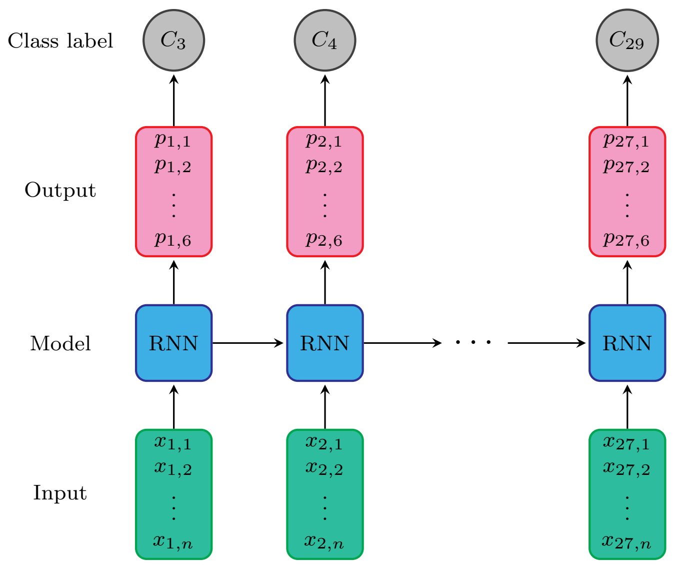

Class label
C_3
C_4
C_{29}
Output
$p_{1,1}$ $p_{1,2}$ $p_{1,6}$
$p_{2,1}$ $p_{2,2}$ $p_{2,6}$
$p_{27,1}$ $p_{27,2}$ $p_{27,6}$
Model
RNN
RNN
RNN
Input
$x_{1,1}$ $x_{1,2}$ $x_{1,n}$
$x_{2,1}$ $x_{2,2}$ $x_{2,n}$
$x_{27,1}$ $x_{27,2}$ $x_{27,n}$

Fig. 12 Graphical depiction of the alternative RNN model using a value of $k = 2$. The variable $x_{i,j}$ represents the j-th input feature at question i, where the total number of features, n, is equal to 1884. The value of $p_{i,j}$ in the output represents the predicted probability of class label j at question i. The target class label at question i is the ALEKS PPL course placement recommendation C_{i+k} at question $i+k$. While our previous RNN models always used C_{29} as the target label at each step in the sequence, in this model the target label changes based on the question number. In this example with $k = 2$, the target label at question i is equal to the ALEKS PPL course placement recommendation from question $i+2$.

more, we do not need to make any additional changes to our RNN model, or to Algorithm 1 (other than adjusting the thresholds, which we discuss shortly); once we have trained our RNN model, the application of the stopping algorithm works as before. Figure 12 contains a graphical representation of this new model.

The intuition behind this approach is the following. Suppose the assessment is currently at question n. If the RNN model is very confident when predicting the course placement recommendation for question $n + k$, this can be interpreted as evidence that the assessment has stabilized and can be stopped. Although we would expect at least some drop in accuracy in comparison to having access to the course recommendation using all 29 questions, the upside to this approach is that it is easier for us to obtain access to the ground truth labels for a model that has been deployed to production. That is, rather than having to let a subset of assessments run the full 29 questions, we only need to let this subset of assessments continue for an extra k questions, where we assume that k is a small number (under this assumption, it may even be feasible to let *every* assessment continue for an extra k questions); as we will see shortly, $k = 2$ is a reasonable value.

Additionally, with this new approach, since the prediction at question n only needs to be evaluated against the label at question $n + k$, we can still measure the accuracy of the first $N - k$ questions of any assessment of length N, regardless of whether or not the assessment was allowed to run further for data collection purposes. For example, suppose the stopping algorithm decides to end an assessment after question 22, using a value of $k = 2$. This assessment will provide labeled data for questions 1 to 20 that can be used to evaluate the RNN predictions. (This is in contrast to the original approach that always requires the label at question 29 to evaluate the predictions at any of the previous questions.) Alternatively, if we wanted to evaluate the predictions at questions 21 and 22 for the same assessment, we would simply have to ask two additional questions (beyond the decision of the stopping algorithm). Thus, this labeling strategy allows for much greater flexibility when evaluating the performance of the RNN model.

When applying Algorithm 1 with these modified models on our validation set, we observed that the classifiers suffer from low accuracy if the stopping algorithm is applied too early. This isn't all that surprising, as the classifiers are very limited in their scope. For example, using the two-question model (i.e., $k = 2$), at question 10 in the assessment the model is only making a prediction about the student's placement recommendation at question 12, which may not match the final label in many cases. Thus, to compensate for this, we activate the stopping algorithm only for questions 17 and later (this value was tuned on our validation set). Additionally, our experiments on the validation set showed that, while $k = 2$ gave better performance in comparison to $k = 1$, there was little gain from using values larger than $k = 2$; thus, for our evaluation on the test data we use $k = 2$ for both the GRU and logistic regression models.

The results from applying this modified algorithm are shown in Figure 13. The original GRU model, using the labels from the full-length assessments, has the best overall performance. Looking at the average assessment length, it is between two to three questions shorter than the modified models. Of note, however, is that at accuracy values of 0.995 or above, the performance of the GRU model using the $n + 2$ labels is not far behind the logistic regression model using the full set of labels.

As discussed previously, in comparison to the models using the labels from the full-length assessment, these modified models have a large disadvantage from only knowing the information a few questions ahead. That is, during training the models with the full-length assessment labels "know" where the assessment ends, so to speak; this is in sharp contrast to the modified models, which are trained to make short-term predictions. As shown in Table 4, the average assessment lengths for labels 1 and 6 are less than 18; given that this modified stopping rule doesn't become active until question 17, the difference in performance is perhaps not too surprising. Thus, when deciding on an actual implementation of the stopping algorithm, the extra flexibility afforded by this modified approach must be balanced against the superior performance of the original model. We return to these issues in the discussion section.

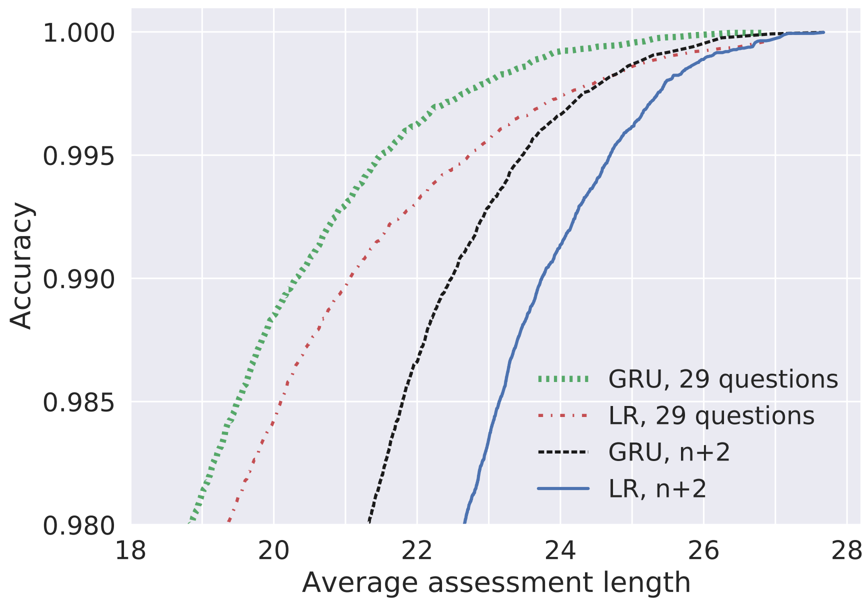


Fig. 13 Accuracy vs. average assessment length on held-out test data for the $n+k$ models, with $k = 2$, and the full models using all 29 questions. All models use the combined set of features.

10 Discussion

In this work we develop and analyze a stopping algorithm for an adaptive assessment. In doing so, we continue the growing trend of applying deep learning and RNN models to educational data. However, in contrast to studies that directly compare the effectiveness of RNN models with more traditional AIED models and techniques, we instead show that augmenting an existing model with an RNN classifier can produce very strong results. When applied to our test data, the RNN stopping algorithm has a substantial shortening effect on the length of the ALEKS PPL assessment, while maintaining the high level of accuracy that is required when offering course placement recommendations. For example, with an accuracy of 0.995 the average number of questions on the assessment is about 21.3, a reduction of over 26% from the full-length of 29 questions.

In addition to looking at the overall accuracy and length statistics, we extend the results from [50] with several additional analyses. For instance, we examine the effect the stopping algorithm has on the time duration of assessments, and we observe a substantial amount of time saved for the typical student. As an example, with an accuracy of 0.995 the median time duration decreases by over 22 minutes, going from 82.5 minutes on the full-length assessments to 60.4 minutes on the shortened assessments; based on this median time for the shortened assessments, we can infer that roughly half of the students would finish the assessment in about an hour or less. Additionally, the stopping algorithm has an even larger effect on the outlier students with very long assessment times, and a closer look at these students reveals some interesting characteristics that require further study. We also evaluate the effect on the final knowledge state returned by the assessment, as in some cases this knowledge state is used as the starting point for a student's remediation and learning in the ALEKS PPL system.

Finally, motivated by the practical issues resulting from the implementation of the stopping algorithm, we introduce and evaluate an alternative model that is trained in a different fashion. While this modified stopping algorithm lags behind the original model in terms of performance, it still results in a large gain over the current 29 question assessment. Furthermore, this modified version is much more flexible when it comes to evaluating its performance and retraining future models. Thus, our current take is that both of these models have their uses, and we imagine a scenario in which, depending on the situation, one or the other model may be deployed. For example, given a large institution with lots of previous ALEKS PPL assessment data, we believe that the use of the model with labels from the full-length assessment would be preferred. In this scenario, the machine learning model can be evaluated on the large set of historical data associated to this institution; while this may not be as ideal as having fully labeled data from subsequent assessments, observing consistent performance on several years of historical data would give us more confidence in the applicability of the model to future student populations. Additionally, a large institution would also generate more external data, such as course grades, that can be used to continually evaluate the accuracy of the course placement predictions.

Now, contrast this with the situation that occurs if an institution is small and has little historical data, or has never used ALEKS PPL previously. In these cases, the lack of existing data would make it difficult to validate the performance of the RNN model on this new student population, as we would not have access to enough labeled data from full-length assessments. Thus, in such situations we would prefer to deploy the modified model with the $n + k$ question labels. Using this model would give us the flexibility to more thoroughly evaluate the performance of the stopping algorithm on this new population, and it would allow us to retrain the classifier and fine-tune it if necessary. Eventually, once we are confident in the performance of the stopping algorithm (say, after a year or two), it's possible the institution could then be moved over to the full model that is more efficient.

There are a couple of directions for further improvements to the stopping algorithm that we are currently exploring. In regards to the aforementioned practical issues associated with the loss of ground truth labels, a promising approach is to use simulated data to make up the deficit. Previous works have used simulated students to analyze the possible effects of changes to productions systems [19,27]. In the specific case of ALEKS PPL, the detailed approach for simulating student responses developed in [22] could be used to generate additional training data for the RNN models.

We are also in the process of exploring the possible benefits of using a different architecture from the RNN models we applied. In particular, the Transformer [75] is a newer architecture that is also designed to handle sequential data. Transformers have surpassed RNN models as the state-of-the-art in many areas of natural language processing [7,18,47,82]. Compared to RNN models, Transformers can be trained faster and more efficiently [75], and they can handle longer sequences of data, with training being possible on sequences of length greater than 12,000 [10].

While our analysis of the Transformer architecture for the current task is very preliminary, we can say that the initial results have not yielded any significant improvements, either in training time or predictive performance. However, while our models and data set are relatively large for the field of AIED, both are substantially smaller than the largest such examples that appear in the more general artificial intelligence field. For example, the largest Transformer model to date has 175 billion parameters [7]; as the largest model we built has roughly 41 million parameters, it is smaller by over a factor of 4000. Additionally, our sequences have length 29, which falls far short of the longest sequences Transformers are capable of handling. If our initial results hold and we do not see a major benefit to applying Transformer models, these large differences in scale may be at play; that is, the major benefits of Transformers may not appear at these smaller sizes.

Acknowledgements We would like to thank the anonymous reviewers for their many helpful suggestions and comments.

References

1. Baker, R.S.: Stupid tutoring systems, intelligent humans. International Journal of Artificial Intelligence in Education **26**(2), 600–614 (2016)
2. Baker, R.S.J.d., Corbett, A.T., Aleven, V.: More accurate student modeling through contextual estimation of slip and guess probabilities in Bayesian Knowledge Tracing. In: Intelligent Tutoring Systems, pp. 406–415. Springer Berlin Heidelberg (2008)
3. Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: A practical and powerful approach to multiple testing. Journal of the Royal Statistical Society: Series B (Methodological) **57**(1), 289–300 (1995). DOI 10.1111/j.2517-6161.1995.tb02031.x. URL `https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1995.tb02031.x`
4. Benjamini, Y., Yekutieli, D.: The control of the false discovery rate in multiple testing under dependency. The Annals of Statistics **29**(4), 1165–1188 (2001). URL `http://www.jstor.org/stable/2674075`
5. Botelho, A., Baker, R., Heffernan, N.: Improving sensor-free affect detection using deep learning. In: Artificial Intelligence in Education-18th International Conference, AIED 2017, pp. 40–51 (2017)
6. Boughorbel, S., Jarray, F., El-Anbari, M.: Optimal classifier for imbalanced data using Matthews correlation coefficient metric. PLOS ONE **12**(6), e0177678 (2017)
7. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. arXiv preprint arXiv:2005.14165 (2020)
8. Cen, H., Koedinger, K., Junker, B.: Is over practice necessary? - Improving learning efficiency with the cognitive tutor through educational data mining. In: Proceedings of the 13th International Conference on Computers in Education, pp. 511–518 (2007)

9. Chicco, D.: Ten quick tips for machine learning in computational biology. BioData mining **10**(1), 35 (2017)
10. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509 (2019)
11. de Chiusole, D., Stefanutti, L., Anselmi, P., Robusto, E.: Stat-Knowlab. assessment and learning of statistics with competence-based knowledge space theory. International Journal of Artificial Intelligence in Education pp. 1–33 (2020)
12. Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. CoRR **abs/1406.1078** (2014). URL `http://arxiv.org/abs/1406.1078`
13. Chollet, F., et al.: Keras. `https://keras.io` (2015)
14. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
15. Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction **4**(4), 253–278 (1994)
16. Cosyn, E., Uzun, H., Doble, C., Matayoshi, J.: A practical perspective on knowledge space theory: ALEKS and its data (2020). Submitted for publication
17. Desmarais, M.C., d Baker, R.S.: A review of recent advances in learner and skill modeling in intelligent learning environments. User Modeling and User-Adapted Interaction **22**(1-2), 9–38 (2012)
18. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186 (2019)
19. Dickison, D., Ritter, S., Nixon, T., Harris, T.K., Towle, B., Murray, R.C., Hausmann, R.G.: Predicting the effects of skill model changes on student progress. In: International Conference on Intelligent Tutoring Systems, pp. 300–302. Springer (2010)
20. Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation **10**(7), 1895–1923 (1998)
21. Ding, X., Larson, E.C.: Why Deep Knowledge Tracing has less depth than anticipated. In: Proceedings of the 12th International Conference on Educational Data Mining, pp. 282–287 (2019)
22. Doble, C., Matayoshi, J., Cosyn, E., Uzun, H., Karami, A.: A data-based simulation study of reliability for an adaptive assessment based on knowledge space theory. International Journal of Artificial Intelligence in Education **29**, 258–282 (2019). DOI 10.1007/s40593-019-00176-0
23. Doignon, J.P., Falmagne, J.C.: Spaces for the assessment of knowledge. International Journal of Man-Machine Studies **23**, 175–196 (1985)
24. Edwards, A.L.: Note on the correction for continuity in testing the significance of the difference between correlated proportions. Psychometrika **13**(3), 185–187 (1948)
25. Falmagne, J.C., Albert, D., Doble, C., Eppstein, D., Hu, X. (eds.): Knowledge Spaces: Applications in Education. Springer-Verlag, Heidelberg (2013)
26. Falmagne, J.C., Doignon, J.P.: Learning Spaces. Springer-Verlag, Heidelberg (2011)
27. Fancsali, S.E., Nixon, T., Vuong, A., Ritter, S.: Simulated students, mastery learning, and improved learning curves for real-world cognitive tutors. In: AIED 2013 Workshops Proceedings Volume 4, p. 11 (2013)
28. Gal, Y., Ghahramani, Z.: A theoretically grounded application of dropout in recurrent neural networks. In: Advances in Neural Information Processing Systems 29 (NeurIPS) (2016)
29. González-Espada, W.J., Bullock, D.W.: Innovative applications of classroom response systems: Investigating students item response times in relation to final course grade, gender, general point average, and high school ACT scores. Electronic Journal for the Integration of Technology in Education **6**, 97–108 (2007)
30. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). `http://www.deeplearningbook.org`
31. Gorodkin, J.: Comparing two k-category assignments by a k-category correlation coefficient. Computational biology and chemistry **28**(5-6), 367–374 (2004)
32. Graves, A., Mohamed, A., Hinton, G.: Speech recognition with deep recurrent neural networks. In: Proceedings of ICASSP 2013, pp. 6645–6649 (2013)

33. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**, 1735–1780 (1997)
34. Hockemeyer, C., Held, T., Albert, D.: RATH-a relational adaptive tutoring hypertext WWW-environment based on knowledge space theory (1997)
35. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 448–456 (2015)
36. Jiang, W., Pardos, Z., Wei, Q.: Goal-based course recommendation. In: Proceedings of the 9th International Conference on Learning Analytics and Knowledge, pp. 36–45 (2019)
37. Jiang, Y., Bosch, N., Baker, R.S., Paquette, L., Ocumpaugh, J., Andres, J.M.A.L., Moore, A.L., Biswas, G.: Expert feature-engineering vs. deep neural networks: which is better for sensor-free affect detection? In: Artificial Intelligence in Education-19th International Conference, AIED 2018, pp. 198–211 (2018)
38. Käser, T., Klingler, S., Gross, M.: When to stop? Towards universal instructional policies. In: Proceedings of the Sixth International Conference on Learning Analytics & Knowledge, pp. 289–298 (2016)
39. Käser, T., Schwartz, D.L.: Exploring neural network models for the classification of students in highly interactive environments. In: Proceedings of the 12th International Conference on Educational Data Mining, pp. 109–118 (2019)
40. Khajah, M., Lindsey, R., Mozer, M.: How deep is knowledge tracing? In: Proceedings of the 9th International Conference on Educational Data Mining, pp. 94–101 (2016)
41. Klingler, S., Käser, T., Busetto, A.G., Solenthaler, B., Kohn, J., von Aster, M., Gross, M.: Stealth assessment in ITS-a study for developmental dyscalculia. In: International Conference on Intelligent Tutoring Systems, pp. 79–89. Springer (2016)
42. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
43. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
44. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**, 436–444 (2015)
45. Lee, J.I., Brunskill, E.: The impact on individualizing student models on necessary practice opportunities. In: Proceedings of the 5th International Conference on Educational Data Mining, pp. 118– 125 (2012)
46. Lin., C., Chi, M.: A comparison of BKT, RNN and LSTM for learning gain prediction. In: Artificial Intelligence in Education-18th International Conference, AIED 2017, pp. 536–539 (2017)
47. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: RoBERTa: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
48. Lynch, D., Howlin, C.P.: Real world usage of an adaptive testing algorithm to uncover latent knowledge. In: Proceedings of the 7th International Conference of Education, Research and Innovation, pp. 504–511
49. Mao, Y., Lin, C., Chi, M.: Deep learning vs. Bayesian Knowledge Tracing: Student models for interventions. Journal of Educational Data Mining **10**(2), 28–54 (2018)
50. Matayoshi, J., Cosyn, E., Uzun, H.: Using recurrent neural networks to build a stopping algorithm for an adaptive assessment. In: Artificial Intelligence in Education-20th International Conference, AIED 2019, pp. 179–184 (2019)
51. Matayoshi, J., Granziol, U., Doble, C., Uzun, H., Cosyn, E.: Forgetting curves and testing effect in an adaptive learning and assessment system. In: Proceedings of the 11th International Conference on Educational Data Mining, pp. 607–612 (2018)
52. Matayoshi, J., Uzun, H., Cosyn, E.: Deep (un)learning: Using neural networks to model retention and forgetting in an adaptive learning system. In: Artificial Intelligence in Education-20th International Conference, AIED 2019, pp. 258–269 (2019)
53. Matthews, B.W.: Comparison of the predicted and observed secondary structure of t4 phage lysozyme. Biochimica et Biophysica Acta (BBA)-Protein Structure **405**(2), 442–451 (1975)
54. McGraw-Hill Education/ALEKS Corporation: What is ALEKS? `https://www.aleks.com/about_aleks` (2019)

55. McNemar, Q.: Note on the sampling error of the difference between correlated proportions or percentages. Psychometrika **12**(2), 153–157 (1947)
56. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
57. Mojarad, S., Essa, A., Mojarad, S., Baker, R.S.: Data-driven learner profiling based on clustering student behaviors: learning consistency, pace and effort. In: International Conference on Intelligent Tutoring Systems, pp. 130–139. Springer (2018)
58. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An imperative style, high-performance deep learning library. In: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019). URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`
59. Pavlik, P.I., Cen, H., Koedinger, K.R.: Performance factors analysis–a new alternative to knowledge tracing. In: Artificial Intelligence in Education-14th International Conference, AIED 2009 (2009)
60. Pavlik, P.I., Olney, A.M., Bankder, A., Eglington, E., Yarbro, J.: The mobile fact and concept textbook system (MoFaCTS). In: Proceedings of the Second Workshop on Intelligent Textbooks, International Conference on Artificial Intelligence in Education (2020)
61. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine Learning in Python . Journal of Machine Learning Research **12**, 2825–2830 (2011)
62. Pelc, A.: Searching games with errors–fifty years of coping with liars. Theoretical Computer Science **270**(1-2), 71–109 (2002)
63. Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L., Sohl-Dickstein, J.: Deep Knowledge Tracing. In: Advances in Neural Information Processing Systems, pp. 505–513 (2015)
64. Powers, D.M.: Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. Journal of Machine Learning Technologies pp. 37–63 (2011)
65. Prechelt, L.: Early stopping – but when? In: G. Montavon, G. Orr, K. Müller (eds.) Neural Networks: Tricks of the Trade, *Lecture Notes in Computer Science*, vol. 7700. Springer, Berlin, Heidelberg (2012)
66. Reddy, A., Harper, M.: Mathematics placement at the University of Illinois. PRIMUS **23**, 683–702 (2013)
67. Rollinson, J., Brunskill, E.: From predictive models to instructional policies. In: Proceedings of the 8th International Conference on Educational Data Mining, pp. 179–186 (2015)
68. Ruseti, S., Dascalu, M., Johnson, A.M., Balyan, R., Kopp, K.J., McNamara, D.S., Crossley, S.A., Trausan-Matu, S.: Predicting question quality using recurrent neural networks. In: International Conference on Artificial Intelligence in Education, pp. 491–502. Springer (2018)
69. Sak, H., Senior, A., Beaufays, F.: Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In: Fifteenth Annual Conference of the International Speech Communication Association (2014)
70. Santurkar, S., Tsipras, D., Ilyas, A., Madry, A.: How does batch normalization help optimization? In: Advances in Neural Information Processing Systems, pp. 2483–2493 (2018)
71. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D.: Mastering the game of Go with deep neural networks and tree search. Nature **529**, 484–503 (2016). URL `http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html`

72. Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al.: Mastering the game of Go without human knowledge. Nature **550**(7676), 354 (2017)
73. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**, 1929–1968 (2014)
74. Theano Development Team: Theano: A Python framework for fast computation of mathematical expressions. arXiv e-prints **abs/1605.02688** (2016). URL `http://arxiv.org/abs/1605.02688`
75. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
76. Wang, Y., Heffernan, N.T.: Leveraging first response time into the knowledge tracing model. In: Proceedings of the 5th International Conference on Educational Data Mining, pp. 176–179 (2012)
77. Wilson, K.H., Karklin, Y., Han, B., Ekanadham, C.: Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation. In: Proceedings of the 9th International Conference on Educational Data Mining, pp. 539–544 (2016)
78. Wilson, K.H., Xiong, X., Khajah, M., Lindsey, R.V., Zhao, S., Karklin, Y., Van Inwegen, E.G., Han, B., Ekanadham, C., Beck, J.E., et al.: Estimating student proficiency: Deep learning is not the panacea. In: Neural Information Processing Systems, Workshop on Machine Learning for Education, p. 3 (2016)
79. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M.: Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv e-prints (2016). ArXiv:1609.08144[cs]
80. Xiong, X., Zhao, S., Vaninwegen, E., Beck, J.: Going deeper with knowledge tracing. In: Proceedings of the 9th International Conference on Educational Data Mining, pp. 545–550 (2016)
81. Xu, L., Davenport, M.: Dynamic knowledge embedding and tracing. In: Proceedings of the 13th International Conference on Educational Data Mining, pp. 524–530 (2020)
82. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. In: Advances in Neural Information Processing Systems, pp. 5754–5764 (2019)
83. Yin, W., Kann, K., Yu, M., Schütze, H.: Comparative study of CNN and RNN for natural language processing. arXiv preprint arXiv:1702.01923 (2017)
84. Yudelson, M.V., Koedinger, K.R., Gordon, G.J.: Individualized Bayesian Knowledge Tracing models. In: Artificial Intelligence in Education-16th International Conference, AIED 2013, pp. 171–180. Springer (2013)