Incorporating Features Learned by an Enhanced Deep Knowledge Tracing Model for STEM/Non-STEM Job Prediction

Chun-Kit Yeung1 · Dit-Yan Yeung2

Published online: 6 May 2019
© International Artificial Intelligence in Education Society 2019

Abstract
The 2017 ASSISTments Data Mining competition aims to use data from a longitudinal study for predicting a brand-new outcome of students which had never been studied before by the educational data mining research community. Specifically, it facilitates research in developing predictive models that predict whether the first job of a student out of college belongs to a STEM (the acronym for science, technology, engineering, and mathematics) field. This is based on the student’s learning history on the ASSISTments blended learning platform in the form of extensive clickstream data gathered during the middle school years. To tackle this challenge, we first estimate the expected knowledge state of students with respect to different mathematical skills using a deep knowledge tracing (DKT) model and an enhanced DKT (DKT+) model. We then combine the features corresponding to the DKT/DKT+ expected knowledge state with other features extracted directly from the student profile in the dataset to train several machine learning models for the STEM/non-STEM job prediction. Our experiments show that models trained with the combined features generally perform better than the models trained with the student profile alone. Detailed analysis on the student’s knowledge state reveals that, when compared with non-STEM students, STEM students generally show a higher mastery level and a higher learning gain in mathematics.

Keywords Educational data mining · Career prediction · Knowledge tracing · Machine learning

More information can be found on http://sites.google.com/view/assistmentsdatamining/.

This paper is based on a method that received the first place in the 2017 ASSISTments data mining competition. The implementation of this work is available on https://github.com/ckyeungac/ADM2017.

Chun-Kit Yeung
chunkit@findsolutiongroup.com
https://ckyeungac.com

Extended author information available on the last page of the article.
Introduction

With the advances and prevalence in online learning platforms such as intelligent tutoring systems (ITSs) and massive open online courses (MOOCs), these platforms have been producing massive amount of educational data. The types of data range from raw activity log files to information captured by a camera or some kind of sensor. The availability of rich data sources has accelerated the development of learning analytics and educational data mining tools. For example, some of the work has been focusing on pedagogical recommendation (Mi and Faltings 2017; Zhou et al. 2017), measurements of student behaviors in ITSs (Reddy et al. 2016; Geigle and Zhai 2017; Klingler et al. 2017), and the detection of learning behaviors in a physical environment, such as wandering detection (Stewart et al. 2017). Some other work has concentrated on the student performance prediction, such as dropout prediction (Yang et al. 2013; Fei and Yeung 2015), grade prediction (Ashenafi et al. 2015; Ren et al. 2017), and knowledge tracing (Corbett and Anderson 1995; Pavlik et al. 2009; Piech et al. 2015; Zhang et al. 2017).

Besides the work from the research community, data mining competitions also attract data science experts to improve the quality of education with the use of data regarding student’s learning. ASSISTments Data Mining competition (ADM) 2017 is one of them and has attracted more than 70 teams across the globe to develop models that predict whether the student’s first job out of college belongs to a STEM/non-STEM field. This is done by exploiting the extensive clickstream data collected from the ASSISTments blended learning platform during the middle school years. Although a similar work (San Pedro et al. 2014) has been done to predict whether a student will choose a STEM major in university, this task has scarcely been seen in published research before, and hence the ADM 2017 has shed light on a new application in educational data mining and learning analytics.

Tackling the ADM 2017, we conduct an analysis on its dataset and discover important factors that potentially influence the student’s first job to be a STEM/non-STEM field. It is observed that the student’s mastery level, or so-called knowledge state, of mathematics has the potential power to distinguish between STEM and non-STEM classes. Accordingly, we suggest using the state-of-the-art knowledge tracing algorithm, deep knowledge tracing (DKT) (Piech et al. 2015), and an enhanced DKT (DKT+) (Yeung and Yeung 2018), to estimate the expected student’s knowledge state on different cognitive skills in mathematics. The expected knowledge state is then combined with the student profile in the ADM 2017 dataset to train a predictive model that classifies whether a student will choose a STEM field in his first job. Apart from logistic regression (LR) used in San Pedro et al. (2014), gradient boosted decision tree (GBDT), linear discriminant analysis (LDA) and support vector machine (SVM) are employed to tackle the ADM 2017.

Our main contributions are summarized as follows:

• DKT/DKT+ model is applied in STEM/non-STEM job prediction. Our experiments show that models trained with the combined features, i.e., the student’s expected knowledge and the student profile, generally perform better than the student profile alone.
Detailed analysis is conducted on the relationship between knowledge state and the STEM class. It reveals that STEM students generally show a higher mastery level and higher learning gain in mathematics, compared with non-STEM students.

**ADM 2017 Dataset**

This dataset comes from a longitudinal study which tracks students from their use of the ASSISTments blended learning platform\(^1\) in the middle school during 2004-2007, to their high school course-taking, college enrollment, and the first job out of college. Despite that substantial student information is collected in this longitudinal study, this competition only provides students’ clickstream records during their middle school year on the ASSISTments system and the STEM labels that have been collected in the longitudinal study.

The dataset consists of the clickstream records from 1,709 students with 942,816 interactions in total when they were using the ASSISTments system. Although 1,709 students are presented in the dataset, only 467 of them are given the STEM/non-STEM label. Among them, 117 students belong to the STEM class and the remaining 350 students belong to the non-STEM class. These numbers are summarized in Table 1.

**Data Description**

The dataset provides two types of data – clickstream records and student profiles. A clickstream record is generated whenever a student answers a question in the ASSISTments system, while a student profile describes background information of the student and summarizes his clickstream record. In Table 2, an example of a student profile and a subset of clickstream record is shown.

**Clickstream record**

Each clicksteam record contains information about the student usage with the system to some learning indicators, in total comprising 64 attributes. For example, the record contains “correct” and “hint” which indicate whether the student answered the question correctly and used hint(s) in the interaction, respectively; “attemptCount” and “hintCount” indicate the number of attempts made and the number of hints used, respectively, so far to solve the question; and “timeTaken” records the time that the student spent in second. Not only is the student usage provided, but also information about the question, such as “problemId” providing a unique identifier of the

---

\(^1\)The ASSISTments system is a tutoring system which provides mathematical questions for students, aligned with their school classes. Each question in the system is associated with at least one mathematical cognitive skill, such as “addition” and “subtraction”. When a student answers a question on the ASSISTments system, it generates clickstream records about the learning behavior of the student from several educational aspects.
Table 1  Summary of the ADM 2017 dataset

| Number of  | #  |
|------------|----|
| Clickstream records | 942,816 |
| Students with no label | 1,242 |
| Students with non-STEM label | 350 |
| Students with STEM label | 117 |

question being answered, “scaffold” indicating whether the question is a scaffolding question, and “skill” revealing the cognitive skill associated with the question are provided. In addition, there are some learning indicators representing the students’ learning status in each interaction. First, the system estimates the student’s knowledge state (“Ln”) using the BKT algorithm. Each time a student answers a question, the ASSISTments system recalculates the student’s knowledge state of the cognitive skill associated with the question being answered. Moreover, student’s affect behaviors (Ocumpaugh et al. 2014) are detected, such as confusion (“RES_CONFUSED”), frustration (“RES_FRUSTRATED”), boredom (“RES_BORED”), and engaged concentration (“RES_CONCENTRATING”). Furthermore, student’s disengaged behaviors, such as, off-task behavior (“RES_OFFTASK”) (Baker 2007) and gaming the system (“RES_GAMING”) (Baker et al. 2004) are modeled in each interaction.

**Student Profile**

The student profile has 11 attributes, providing the student’s background information and summarizing the student usage of the ASSISTments system and the learning indicators mentioned above. One of them is “SY ASSISTments Usage” which marks the school year when the student used the ASSISTments system. Another one is “NumActions” which is the number of interactions that the student had with the ASSISTments system. The remaining 9 attributes are the summary of the student’s clickstream record, and they can be divided into the following three categories:

1. **Student ability**: This includes “AveKnow” and “AveCorrect”, representing the averaged last knowledge state of each mathematical skill that the student has already answered, and the correct rate of the student.

2. **Affective states**: This includes “AveResConf”, “AveResFrust”, “AveResBored”, and “AveResEngcon”, representing the averaged tendencies of confusion, frustration, boredom and engaged concentration.

3. **Disengaged behaviors**: This includes “AveCarelessness” (San Pedro et al. 2011), “AveResGaming”, and “AveResOfftask”, representing the averaged tendencies of the student slipping a question, gaming the system and disengaging from the system.

**Analyzing Student Profile**

Although the clickstream data provides profuse information about students’ learning at each interaction, it simultaneously poses a challenge in analyzing the relationship
among their temporal changes and their career choice. Therefore, we conduct an analysis on the student profile, instead of the clickstream data, to discover potential factors that can distinguish between STEM and non-STEM classes.

Independent sample t-tests on each attribute in the student profile between STEM and non-STEM classes are conducted for the sake of testing whether there is a statistically significant difference between the two classes in terms of their mean values.²

²Note that “SY ASSISTments Usage” is excluded in the t-test.
The obtained result is shown in Table 3, which presents a similar discovery in the previous work conducted by San Pedro et al. (2014).

The result demonstrates that the non-STEM class has higher mean values on “NumActions”, “AveResFrustr” and “AveResGaming”, whereas the STEM class has higher mean values on the “AveCarelessness”, “AveKnow” and “AveCorrect”. Furthermore, “AveKnow”, “AveCorrect” and “AveCarelessness” seem to be potential predictors, as they have a large magnitude in t-score and Cohen’s d as well as a small p-value, indicating the STEM and the non-STEM classes have different mean values in these attributes with a high confidence level. One of the possible reasons is that students who are proficient in mathematics are better at dealing with extensive mathematical reasoning. Therefore, it is easier for them to discover interest in the STEM field which profoundly involves mathematics. As for the reason why “AveCarelessness” becomes a discriminating variable in the t-test, it might be attributed to the phenomena that sloppiness in tackling problems is more common in academically outstanding students (Clements 1982). In contrast, “AveResBored” and “AveResConf” are less effective at discriminating STEM and non-STEM students. It is plausibly because a question’s content plays a more significant role in influencing students’ emotions rather than the students’ characteristics. Hence, the values of students’ affective states mostly reflect the traits of a question, which will not be paramount factors in deciding whether a student will pursue a career in the STEM field. Therefore, the features which indicate students’ ability are displayed more distinctly between the STEM and the non-STEM classes, while the affective states are almost identical between the two classes.

Table 3  The result of the independent sample t-tests on student profile. Bonferoni corrected p-values are also reported as “Adjusted p-value”

| Attribute       | t-score | p-value | Adjusted p-value | Cohen’s d STEM avg | std | STEM non-STEM avg | std |
|-----------------|---------|---------|------------------|-------------------|-----|-------------------|-----|
| AveCarelessness | −4.1934 | 0.0000  | 0.0004           | −0.4833           | 0.1544 | 0.0791           | 0.1201 | 0.0670 |
| AveKnow         | −4.0466 | 0.0001  | 0.0008           | −0.4704           | 0.2923 | 0.1622           | 0.2245 | 0.1378 |
| AveCorrect      | −4.0159 | 0.0001  | 0.0009           | −0.4560           | 0.4796 | 0.1442           | 0.4191 | 0.1285 |
| AveResGaming    | 2.5690  | 0.0108  | 0.0180           | 0.2475            | 0.0822 | 0.1019           | 0.1120 | 0.1264 |
| AveResFrustr    | 1.6633  | 0.0979  | 0.9794           | 0.1865            | 0.1195 | 0.0531           | 0.1288 | 0.0485 |
| AveResEngcon    | −1.2429 | 0.2151  | 1.0000           | −0.1210           | 0.6514 | 0.0251           | 0.6478 | 0.0304 |
| NumActions      | 0.6992  | 0.4851  | 1.0000           | 0.0691            | 518.2564 | 344.5232       | 545.2914 | 405.7526 |
| AveResOfftask   | −0.2083 | 0.8352  | 1.0000           | −0.0209           | 0.2203 | 0.0757           | 0.2185 | 0.0859 |
| AveResConf      | 0.0848  | 0.9325  | 1.0000           | 0.0087            | 0.1058 | 0.0365           | 0.1061 | 0.0398 |
| AveResBored     | −0.0359 | 0.9714  | 1.0000           | −0.0037           | 0.2530 | 0.0316           | 0.2529 | 0.0346 |
Model Formulation

As the student’s ability in mathematics has the potential to distinguish between the STEM and the non-STEM classes, we believe that a comprehensive student’s knowledge state is more conducive than the average knowledge state (“AveKnow”) provided in the dataset, which is merely a single value. Thus, it is desirable for the knowledge state of each mathematical skill to be the features of the predictive model. Yet, to obtain a comprehensive knowledge state of a student, DKT is a better option than BKT because BKT is only capable of estimating the knowledge state of skills that the student has already answered. For those skills that have never been answered by the student, BKT cannot be used to estimate their knowledge state. Hence, missing values will eventually result if BKT is used to extract the student’s expected knowledge state of each skill. Although these missing values could be filled by $P(L_0)$ which is the initial probability of knowing a KC in the BKT model, these values are anticipated to be similarly small in magnitude, so students’ mathematical ability on those unseen skills may be underestimated.

To address the above issue, we adopt the knowledge tracing (KT) algorithm DKT, which estimates the knowledge state of all the skills simultaneously and outperforms many of the traditional KT models such as BKT and performance factor analysis (PFA) (Pavlik et al. 2009) without much need of feature engineering. As stated by Khajah et al. (2016), DKT is capable of capturing the recency effect, contextualized trial sequence, inter-skill relationship and students’ ability merely from question-and-answer interactions. Therefore, we propose to combine the DKT knowledge state and the student profile to be the feature set of the predictive model. In addition to the DKT model, we also adopt the enhanced DKT which resolves two deficits discovered in the DKT model.

Enhanced Deep Knowledge Tracing (DKT+)

Concretely, the KT task can be formalized as follows: given a student’s historical interactions $X_t = (x_0, x_1, \ldots, x_t)$ on a particular learning task, it predicts some aspects of the student’s next interaction $x_{t+1}$. Question-and-answer interaction is the most common type of interaction in KT, and is usually represented as an ordered pair $(q_t, a_t)$ where $q_t$ is the question being answered at time $t$ and $a_t$ is the answer label indicating whether the question has been answered correctly. With respect to this setting, KT predicts the probability that the student will correctly answer the next question $q_{t+1}$.

DKT employs the recurrent neural network (RNN) (Lipton et al. 2015) with the long short-term memory (LSTM) cells (Olah 2015) as its backbone model. In DKT, an interaction $(q_t, a_t)$ needs to be transformed to a fixed-length input vector $x_t$. A question $q_t$ is represented in form of one-hot encoding to form a vector $\delta(q_t)$ since the question can be identified by a unique ID. The corresponding answer label can also be represented as the one-hot vector $\delta(q_t)$ if a student answers $q_t$ correctly, or a zero
vector $\mathbf{0}$ otherwise. Therefore, if there are $M$ unique questions, then $\mathbf{x}_t \in \{0, 1\}^{2M}$. After the transformation, DKT passes the $\mathbf{x}_t$ to the LSTM-RNN to compute the output vector $\mathbf{y}_t$ which represents the probabilities of answering each question correctly. For student $i$, if he has a sequence of question-and-answer interactions of length $T_i$, the DKT model maps the inputs $(\mathbf{x}_{i1}, \mathbf{x}_{i2}, \ldots, \mathbf{x}_{iT_i})$ to the outputs $(\mathbf{y}_{i1}, \mathbf{y}_{i2}, \ldots, \mathbf{y}_{iT_i})$ accordingly. Moreover, since the objective of the DKT is to predict the next interaction performance, so the target prediction of the next question can be extracted by performing a dot product of the output vector $\mathbf{y}_t \cdot \delta(q_{t+1}^i)$ and the one-hot encoded vector of the next question $\delta(q_{t+1}^i)$. Based on the predicted output $\mathbf{y}_t \cdot \delta(q_{t+1}^i)$ and the target output $a_{t+1}^i$, the loss function $L$ can be expressed as follows:

$$L = \frac{1}{\sum_{i=1}^n(T_i - 1)} \left( \sum_{i=1}^n \sum_{t=1}^{T_i-1} l \left( \mathbf{y}_t \cdot \delta(q_{t+1}^i), a_{t+1}^i \right) \right)$$

where $n$ is the number of students in the dataset and $l(\cdot)$ is the cross-entropy loss.

However, as described in Yeung and Yeung (2018), there are two major problems in the existing DKT model. First, it fails to reconstruct the observed question-and-answer interaction, and second, the predicted knowledge state across time-steps is not consistent. Both of the problems are undesirable and unreasonable because student’s knowledge state is expected to transit gradually over time, so we also adopt the DKT+ proposed in Yeung and Yeung (2018), to estimate the knowledge state of students. The DKT+ augments the loss function in (1) with three regularization terms

$$r = \frac{1}{\sum_{i=1}^n(T_i - 1)} \left( \sum_{i=1}^n \sum_{t=1}^{T_i-1} l \left( \mathbf{y}_t \cdot \delta(q_{t+1}^i), a_{t+1}^i \right) \right),$$

$$w_1 = \frac{\sum_{i=1}^n \sum_{t=1}^{T_i-1} \|\mathbf{y}_{t+1}^i - \mathbf{y}_t^i\|_1}{M \sum_{i=1}^n(T_i - 1)}$$

and

$$w_2 = \frac{\sum_{i=1}^n \sum_{t=1}^{T_i-1} \|\mathbf{y}_{t+1}^i - \mathbf{y}_t^i\|_2^2}{M \sum_{i=1}^n(T_i - 1)}$$

where $r$ is the loss between the predicted output and the target output of the current interaction, and $w_1$ and $w_2$ are waviness measures that quantify how disparate the two prediction vectors $\mathbf{y}_t^i$ and $\mathbf{y}_{t+1}^i$ are. Introducing these three regularization terms to the loss function, the model will adjust the prediction with respect to the current input and smoothen the transition in prediction. The new loss function $L’$ of the DKT+ is

$$L’ = L + \lambda_r r + \lambda_w^1 w_1 + \lambda_w^2 w_2^2$$

where $\lambda_r$, $\lambda_w^1$ and $\lambda_w^2$ are regularization parameters.

Concerns of Applying the DKT/DKT+ Model

There are three decisions that should be made when training DKT model and feeding the DKT knowledge state to a machine learning model.
1. Due to both the question-level tag and the skill-level tag of a question being available in the clickstream, we have to decide which tag level should be used when training the DKT model.

2. Owing to different learning trajectories of students, a variable output sequence lengths is resultant from the DKT model. Variable sequence length would be problematic if the machine learning model is static, so we have to handle this issue when feeding the knowledge state into a static machine learning model.

3. As the student knowledge state varies across time-steps, there are numerous ways to select the predicted knowledge state. Should we choose a knowledge state at any arbitrary time-step, or should we aggregate the knowledge state?

Addressing the first point above, the skill-level tag is adopted in this paper, because using the question-level tag will induce sparsity in both vector representation and data density. Hence, using the skill-level tag would increase the accuracy of the DKT model, and thus the output from the DKT model would be more robust when it is used in other tasks.

As for the second and the third points, we decide to extract the knowledge state in the last interaction as the features fed into the machine learning model. It is because the knowledge state computed from the DKT retains the information from past interactions thanks to the RNN’s architecture. Consequently, the last knowledge state $y_T$ embraces the latest student’s mastery level of each mathematical skill. Moreover, using the last knowledge state $y_T$ ensures that the feature size is the same regardless of the number of interactions, so the knowledge state can be fed into a static machine learning model.

**STEM/non-STEM Job Predictor**

After addressing the three points above, we can formulate our prediction model. The student’s last knowledge state $y_T$ (denoted as $x_{KT}$ thereafter for clarity) is combined with the student profile (denoted as $x_{SP}$) described in Table 3 to form the feature set $x_f = [x_{SP}, x_{KT}]$, where $[-]$ is the concatenation operator. Then, machine learning models are used to learn the mapping between $x_f$ and the STEM label $l \in \{0, 1\}$ where 0 and 1 indicate the non-STEM class and the STEM class, respectively. Four popular machine learning models are adopted in this paper to tackle the ADM 2017.

**Gradient Boosted Decision Tree (GBDT)** GBDT is a machine learning technique for regression and classification problems. At each training iteration, GBDT create a new (weak) decision tree to minimize the loss function by exploiting the gradient descent approach. Hence, GBDT is actually a boosting algorithm and generally performs better than a linear model.

**Linear Discriminant Analysis (LDA)** LDA is a method for dimensionality reduction by finding a linear combination of features that characterizes or separates two classes, and the resulting linear combination can also be used for classification. More concretely, LDA makes a prediction by maximum a posteriori probability.
In LDA, the likelihood for each class is assumed to be a multivariate normal distribution parameterized using a mean vector and a covariance matrix. The covariance matrix is assumed to be the same among all classes, and therefore leads to a linear decision boundary.

**Logistic Regression (LR)** LR is one of the most basic machine learning algorithms widely used in the rudimentary state of a machine learning project since it produces good classification accuracy if the learning task is linearly separable. It has one dependent variable and multiple independent variables. The dependent variable is calculated via a logistic function with a set of parameters and the input independent variables.

**Support Vector Machine (SVM)** SVM is one of the most robust algorithms to draw the decision boundary by maximizing the margin between two classes. It can efficiently perform a non-linear classification using the kernel function to implicitly map the inputs into a high-dimensional feature space.

These machine learning models would provide a good baseline and an indicator of the future direction that should be investigated further. All in all, the final architecture of the prediction model is shown in Fig. 1. In the coming chapter, the proposed STEM job predictor, as well as the DKT+ model itself, are evaluated.

**Experiments**

In this experiment, we compare five different sets of features fed into machine learning models. The first set of features is the student profile \( x_{SP} \) alone. It aims to provide

![Fig. 1 The architecture of the STEM/non-STEM job prediction model. It combines the student’s last knowledge state \( y_T \) and the student profile \( x_{SP} \) as the input features](image-url)
a baseline for this experiment. The second and the third sets of features are the student expected knowledge state $x_{KT}$ extracted from DKT and DKT+ respectively. The fourth and the fifth sets of features are the combined features $x_f$, that is, both the student profile $x_{SP}$ and the knowledge state $x_{KT}$. The difference of the fourth and the fifth feature sets is that the DKT knowledge state is used in the fourth feature set, while the DKT+ knowledge state is used in the fifth feature set. We name the models trained with the first feature set with suffix “-SP”, that is, GBDT-SP, LDA-SP, LR-SP and SVM-SP; the models trained with the second and the third feature sets with suffix “-DKT” and “-DKT+”, e.g., LR-DKT and LR-DKT+; and the models trained with the fourth and the fifth feature sets with suffix “-DKT&SP” and “-DKT+&SP” respectively, e.g., LR-DKT&SP and LR-DKT+&SP.

**Implementation**

**Evaluation Measures**

The AUC and the root-mean-square error (RMSE) are adopted as evaluation measures in ADM 2017. AUC provides a robust metric for binary prediction evaluation. When interpreting the value of AUC, the larger the AUC score is, the better the prediction performance is of the model. As for RMSE, it measures the square error between the prediction and the ground truth label. Accordingly, the closer the predictions are made to the ground truths, the smaller the value is of RMSE. In order to have a prediction model with a high AUC score and a small RMSE, the ADM 2017 evaluates the prediction model by a combined score $AUC + (1 - RMSE)$.

Apart from the measures employed in ADM 2017, we adopt the average precision (AP) score to evaluate the model performance. This is because the dataset, where there are 117 STEM students and 350 non-STEM students, is imbalanced. When the dataset is imbalanced, the precision-and-recall (PR) curve is more informative than the AUC metric (Davis and Goadrich 2006). AP is usually reported since it is a single number used to summarize the PR curve. An AP of value 0.5 indicates that, in every ten samples which are predicted to be positive class, five of them are indeed positive. A higher AP score therefore indicates a more accurate binary classifier.

**Experiment Settings**

The training process of the prediction model in this experiment is two-fold. First, a DKT/DKT+ model is trained using all the clickstream data provided in the ADM 2017 dataset, including 1,709 students and 942,816 question-and-answer interactions. Experiment settings and hyperparameter settings for the DKT and DKT+ models are the same as the one reported in Yeung and Yeung (2018). The weights of the model are initialized randomly from a Gaussian distribution with zero mean and small variance. A single-layer RNN-LSTM with a state size of 200 is used as the basis of the DKT and the DKT+ model. The learning rate and the dropout rate are set to 0.01 and 0.5 respectively, with a norm clipping threshold to 3.0. The regularization parameters $\lambda_r$, $\lambda_{w1}$ and $\lambda_{w2}$ for DKT+ are set to 0.1, 0.3 and 3.0 respectively. During training, we use a batch size of 32 and unroll the RNN by the longest sequence’s
length in the batch. All of the sequences in the batch are padded with zero vectors to fill up the remaining time steps. Masking is also applied when computing the loss.

With the trained DKT or DKT+ models, we obtain the knowledge states for the 467 students in the training set. Then, we train and evaluate the four machine learning models with the five feature sets mentioned above. For each model, we perform a hyperparameter grid search and select the hyperparameter setting that results in the highest test combined score (AUC + 1-RMSE) during 5-fold cross-validation.

Hyperparameter Search

Regarding to the GBDT, we vary the number of decision trees (called n_estimators in scikit-learn\(^3\)) in the set of values \{10, 25, 50, 120, 300\} for the model. The maximum depth of each decision tree (called max_depth) and the minimum number of samples required to be a leaf node in the decision tree (called min_sample_leaf) are considered as well. We vary the maximum depth in \{2, 3, 5, 8\} and the minimum number of samples in the leaf node in \{1, 2, 5, 10\}.

Concerning LDA, three different LDA solvers are compared, including singular value decomposition (SVD), least squares solution (LSQR), and eigenvalue decomposition (EIGEN).

As for LR, we search for optimal settings for the regularization parameter \(C\) as well as the regularization strategy, \(L_1\) or \(L_2\) penalty. The value of \(C\) represents the inverse of the regularization strength, so a small value of \(C\) implies a simple model is desired. The value of \(C\) is searched in the set of values \{0.001, 0.01, 0.1, 1.0, 10.0, 100.0\}.

We adopt the radial basis function (RBF) kernel for the SVM and only vary the hyperparameter \(C\) in the set of values \{0.001, 0.01, 0.1, 1.0, 10.0, 100.0\}. The \(C\) in the SVM differs from the \(C\) in the LR, but they are similar. In SVM, the value of \(C\) trades off the misclassification of training samples against the simplicity of the decision surface. A small value of \(C\) indicates a high-tolerance in making a wrong prediction, implying that a simple model is desired.

Results

The experiment result is summarized in Table 4, where the training and the test results are reported in Table 4a and b, respectively. The hyperparameter settings adopted for each model are recorded in Table 5. As for the test result, the models trained with the DKT+ expected knowledge state or the combined features DKT+&SP generally perform better than other features among the models of the same machine learning algorithm, despite the LDA. Among all the models, LR-DKT+&SP achieved the highest combined score of 1.191 and AUC of 0.623. It also gives a RMSE of 0.432 and an AP score of 0.378. SVM-DKT+ results in the highest AP of 0.394, while SVM-DKT+&SP results in the lowest RMSE of 0.429. The result indicates that the expected knowledge state extracted from the DKT+ model is better than that from

\(^{3}\)Scikit-learn documentation can be found on [https://scikit-learn.org/](https://scikit-learn.org/).
Table 4  The training and the test results on the evaluation measures of each model

| Model     | AP      | AUC     | RMSE     | AUC+(1-RMSE) |
|-----------|---------|---------|----------|--------------|
| GBDT-SP   | 0.674 ± 0.038 | 0.833 ± 0.015 | 0.388 ± 0.002 | 1.445 ± 0.017 |
| GBDT-DKT  | 0.970 ± 0.024 | 0.986 ± 0.009 | 0.317 ± 0.020 | 1.669 ± 0.025 |
| GBDT-DKT+ | 0.981 ± 0.023 | 0.995 ± 0.004 | 0.291 ± 0.024 | 1.704 ± 0.028 |
| GBDT-DKT&SP | 0.947 ± 0.032 | 0.975 ± 0.015 | 0.315 ± 0.011 | 1.661 ± 0.026 |
| GBDT-DKT+&SP | **1.000 ± 0.000** | **1.000 ± 0.000** | **0.095 ± 0.020** | **1.905 ± 0.020** |
| LDA-SP    | 0.389 ± 0.031 | 0.668 ± 0.021 | 0.421 ± 0.004 | 1.247 ± 0.024 |
| LDA-DKT   | 0.686 ± 0.027 | 0.846 ± 0.018 | 0.357 ± 0.008 | 1.489 ± 0.026 |
| LDA-DKT+  | 0.708 ± 0.040 | 0.849 ± 0.024 | 0.353 ± 0.010 | 1.496 ± 0.033 |
| LDA-DKT&SP | 0.730 ± 0.026 | 0.864 ± 0.016 | 0.343 ± 0.007 | 1.521 ± 0.023 |
| LDA-DKT+&SP | 0.739 ± 0.038 | 0.867 ± 0.013 | 0.339 ± 0.013 | 1.528 ± 0.025 |
| LR-SP     | 0.381 ± 0.034 | 0.656 ± 0.021 | 0.422 ± 0.004 | 1.234 ± 0.025 |
| LR-DKT    | 0.465 ± 0.017 | 0.697 ± 0.023 | 0.413 ± 0.003 | 1.284 ± 0.026 |
| LR-DKT+   | 0.695 ± 0.045 | 0.844 ± 0.026 | 0.356 ± 0.013 | 1.488 ± 0.038 |
| LR-DKT&SP | 0.607 ± 0.026 | 0.797 ± 0.028 | 0.382 ± 0.009 | 1.415 ± 0.036 |
| LR-DKT+&SP | 0.585 ± 0.042 | 0.796 ± 0.018 | 0.387 ± 0.007 | 1.409 ± 0.023 |
| SVM-SP    | 0.282 ± 0.062 | 0.508 ± 0.058 | 0.448 ± 0.045 | 1.060 ± 0.073 |
| SVM-DKT   | 0.364 ± 0.063 | 0.576 ± 0.094 | 0.432 ± 0.004 | 1.144 ± 0.097 |
| SVM-DKT+  | 0.544 ± 0.046 | 0.727 ± 0.033 | 0.414 ± 0.013 | 1.313 ± 0.036 |
| SVM-DKT&SP | 0.306 ± 0.063 | 0.539 ± 0.080 | 0.434 ± 0.004 | 1.105 ± 0.083 |
| SVM-DKT+&SP | 0.426 ± 0.025 | 0.672 ± 0.012 | 0.422 ± 0.006 | 1.250 ± 0.015 |

(b) Test result

| Model     | AP      | AUC     | RMSE     | AUC+(1-RMSE) |
|-----------|---------|---------|----------|--------------|
| GBDT-SP   | 0.317 ± 0.078 | 0.530 ± 0.079 | 0.438 ± 0.008 | 1.092 ± 0.085 |
| GBDT-DKT  | 0.322 ± 0.106 | 0.550 ± 0.122 | 0.441 ± 0.025 | 1.109 ± 0.146 |
| GBDT-DKT+ | **0.379 ± 0.080** | **0.608 ± 0.026** | **0.430 ± 0.011** | **1.178 ± 0.034** |
| GBDT-DKT&SP | 0.306 ± 0.089 | 0.540 ± 0.107 | 0.449 ± 0.029 | 1.091 ± 0.134 |
| GBDT-DKT+&SP | 0.361 ± 0.068 | 0.587 ± 0.090 | 0.455 ± 0.036 | 1.132 ± 0.115 |
| LDA-SP    | 0.351 ± 0.103 | 0.594 ± 0.080 | 0.434 ± 0.020 | 1.160 ± 0.097 |
| LDA-DKT   | 0.310 ± 0.097 | 0.536 ± 0.080 | 0.501 ± 0.029 | 1.035 ± 0.100 |
| LDA-DKT+  | 0.347 ± 0.098 | 0.559 ± 0.117 | 0.483 ± 0.041 | 1.076 ± 0.158 |
| LDA-DKT&SP | 0.324 ± 0.067 | 0.546 ± 0.097 | 0.504 ± 0.025 | 1.042 ± 0.112 |
| LDA-DKT+&SP | 0.344 ± 0.070 | 0.552 ± 0.083 | 0.489 ± 0.019 | 1.063 ± 0.102 |
| LR-SP     | 0.347 ± 0.123 | 0.595 ± 0.109 | 0.431 ± 0.021 | 1.164 ± 0.129 |
| LR-DKT    | 0.321 ± 0.038 | 0.545 ± 0.081 | 0.434 ± 0.013 | 1.111 ± 0.093 |
| LR-DKT+   | 0.369 ± 0.134 | 0.605 ± 0.089 | 0.464 ± 0.048 | 1.141 ± 0.135 |
| LR-DKT&SP | 0.341 ± 0.110 | 0.582 ± 0.153 | 0.442 ± 0.036 | 1.141 ± 0.187 |
| LR-DKT+&SP | 0.378 ± 0.096 | **0.623 ± 0.100** | 0.432 ± 0.021 | **1.191 ± 0.118** |
| SVM-SP    | 0.311 ± 0.214 | 0.506 ± 0.176 | 0.450 ± 0.044 | 1.056 ± 0.202 |
Table 4 (continued)

| Model            | AP     | AUC    | RMSE  | AUC+(1-RMSE) |
|------------------|--------|--------|-------|--------------|
| SVM-DKT          | 0.343 ± 0.128 | 0.575 ± 0.133 | 0.432 ± 0.007 | 1.143 ± 0.139 |
| SVM-DKT+         | **0.394 ± 0.106** | **0.593 ± 0.088** | **0.426 ± 0.008** | **1.167 ± 0.095** |
| SVM-DKT&SP       | 0.301 ± 0.128 | 0.532 ± 0.164 | 0.434 ± 0.013 | 1.099 ± 0.175  |
| SVM-DKT+&SP      | 0.383 ± 0.090 | **0.596 ± 0.110** | **0.429 ± 0.008** | **1.168 ± 0.115** |

The average and the standard deviation obtained in 5-fold cross-validation are reported. For each column, the best result is shown in bold, and the underlined result indicates it is the best result among the models of the same machine learning algorithm.

The DKT model, and the DKT+ expected knowledge state is helpful in classifying whether a student belongs to STEM class or not.

However, the result of different evaluation measures differs significantly between the training and the test results in majority of the models when DKT/DKT+ expected knowledge state is incorporated, and this demonstrates an overfitting issue. The overfitting issue may be attributed to the curse of dimensionality, because the training

Table 5 The hyperparameter settings of each model reported in Table 4

| Model     | Params                                                                 |
|-----------|-------------------------------------------------------------------------|
| GBDT-SP   | max_depth: 2, min_samples_leaf: 5, n_estimators: 25                     |
| GBDT-DKT  | max_depth: 3, min_samples_leaf: 2, n_estimators: 25                     |
| GBDT-DKT+ | max_depth: 8, min_samples_leaf: 10, n_estimators: 10                    |
| GBDT-DKT&SP | max_depth: 2, min_samples_leaf: 10, n_estimators: 50                   |
| GBDT-DKT+&SP | max_depth: 3, min_samples_leaf: 5, n_estimators: 120                   |
| LDA-SP    | solver: SVD                                                             |
| LDA-DKT   | solver: SVD                                                             |
| LDA-DKT+  | solver: SVD                                                             |
| LDA-DKT&SP | solver: SVD                                                            |
| LDA-DKT+&SP | solver: SVD                                                            |
| LR-SP     | C: 10.0, penalty: L1                                                   |
| LR-DKT    | C: 1.0, penalty: L2                                                    |
| LR-DKT+   | C: 100.0, penalty: L1                                                 |
| LR-DKT&SP | C: 10.0, penalty: L2                                                   |
| LR-DKT+&SP | C: 10.0, penalty: L2                                                  |
| SVM-SP    | C: 0.001                                                               |
| SVM-DKT   | C: 0.001                                                               |
| SVM-DKT+  | C: 100.0                                                               |
| SVM-DKT&SP | C: 0.001                                                              |
| SVM-DKT+&SP | C: 10.0                                                              |
set consists of only 467 samples while the number of features used is up to 112. Even worse, only around 374 students are actually used for training owing to 5-fold cross-validation, leading to an inevitable bottleneck for the performance of classifier. We therefore examine several methods to address the overfitting issue, and they are presented and discussed in Discussion Section – Tackling Overfitting.

**Discussion**

**Tackling Overfitting**

To deal with the curse of dimensionality and the overfitting issue, several methods have been attempted, including dimensionality reduction and oversampling. For the sake of reducing the dimensionality of the feature space, principal component analysis (PCA) and recursive feature elimination (RFE) have been applied. In addition, a variety of oversampling strategies is attempted to augment the number of samples, such as random oversampling (ROS), synthetic minority oversampling technique (SMOTE) and adaptive synthetic (ADASYN) sampling.

**PCA** The dimensionality of the five feature sets is reduced to a few principal components which retain 95% of the proportion of variance (PoV). Then, the reduced feature set is used to train the model.

**RFE** RFE reduces the dimensionality of a feature set by recursively eliminating the least important feature. Specifically, a model is trained on the initial set of features first and accordingly the importance of each feature (if available) is obtained. Next, the least important feature is pruned from the current set of features. This procedure is recursively repeated on the pruned set until the desired number of features is reached. In this procedure, we deem the desired set of features to be the one that produces the highest combined score, i.e., \( \text{AUC} + (1 - \text{RMSE}) \).

**ROS** ROS increases the sample size by randomly selecting minority class samples with replacement. In our case, we sample the STEM samples such that the number of STEM samples is equal to the number of non-STEM samples.

**SMOTE** SMOTE (Chawla et al. 2002) is a more sophisticated method for oversampling. Instead of sampling, it generates samples. The minority class samples are synthesized as follows: for each sample point \( x_i \) in the minority class, SMOTE computes its \( k \) nearest neighbors of the minority class in the dataset. Then, SMOTE randomly chooses \( r \) neighbors’ samples at random. Next, it creates \( r \) synthetic examples randomly along those lines connecting the \( x_i \) with the \( r \) random nearest neighbors.

**ADASYN** Similar to SMOTE, ADASYN (He et al. 2008) generates minority class samples with a slightly different approach. ADASYN uses a density distribution as a criterion to automatically decide the number of synthetic samples that must be
generated for each minority sample by adaptively changing the weights of different minority samples to compensate for the skewed distribution.

Two Python packages are used to perform the above dimensionality reduction and oversampling methods: scikit-learn is employed for PCA and RFE, and imbalanced-learn\(^4\) is employed for ROS, SMOTE and ADASYN. Moreover, when these methods are applied, the experiment procedure follows the one described in the section Experiments.

Among all the attempted methods, the RFE process is the most effective for relieving the overfitting issue and improving the model performance. The result of each model after applying the RFE is shown in Table 6, and the hyperparameter settings are recorded in Table 7.

It is observed that performing the RFE to reduce the dimensionality of the feature set improves the performance of the models. After applying the RFE, LR-DKT&SP performs the best in all of the evaluation measures. The AP, AUC and the combined score are boosted from 0.341 to 0.458, from 0.582 to 0.694, from 1.141 to 1.280, respectively, and the RMSE is reduced from 0.442 to 0.414. The LR-DKT&SP selects 12 out of 112 features, which are “AveKnow”, “AveCorrect”, “AveCarelessness”, together with the knowledge state of 9 mathematical skills – “application: multi-column subtraction”, “area-concept”, “circle-graph”, “finding-percents”, “meaning-of-pi”, “perimeter”, “statistic”, “surface-area-and-volume” and “unit-conversion”\(^5\). Generally, the models trained with the DKT or DKT+ knowledge state, as well as the combined features, result in a higher performance than the models trained with the student profile alone (Table 8).

The reason why the models trained with DKT+ or DKT+&SP feature sets are sometimes inferior to ones trained with DKT or DKT&SP feature sets for the STEM classification problem would be attributed to the features removed during the RFE process. We summarize the features used by each model in Table 9 and find that, in general, more mathematical skills in the knowledge state are removed in the DKT and the DKT&SP feature sets. This observation might imply that knowledge state obtained from the DKT model contains noise and accordingly is irrelevant in distinguishing between the STEM and the non-STEM classes, so the prediction performance of STEM predictors improves after removing much of DKT expected knowledge state. Moreover, we observe that the GBDT models trained with DKT+ and DKT+&SP feature sets perform better. This would be ascribed to the reason that GBDT is more capable of discovering nonlinear relationship between the DKT+ knowledge state and the STEM label.

Nonetheless, it is observed that the GBDT-based models still suffer from the overfitting issue. Moreover, the hyperparameter $C$ in the LR-based models have high values in general, implying that a simple model is not highly desirable. Hence, we speculate that the decision boundary for STEM and non-STEM classes should be more complex than a linear decision boundary, while simultaneously it should not be complex and highly nonlinear.

---

\(^4\)Imbalanced-learn documentation can be found on https://imbalanced-learn.readthedocs.io/en/.

\(^5\)Refer to Table 8 for all of the available skills.
Table 6  The training and the test results on the evaluation measures of each model where the RFE is applied to reduce the dimensionality of the feature set

| Model                  | AP      | AUC    | RMSE   | AUC+(1-RMSE) |
|------------------------|---------|--------|--------|--------------|
| (a) Training result    |         |        |        |              |
| GBDT-SP                | 0.665 ± 0.063 | 0.835 ± 0.029 | 0.394 ± 0.003 | 1.440 ± 0.030 |
| GBDT-DKT               | 0.968 ± 0.017 | 0.985 ± 0.006 | 0.317 ± 0.015 | 1.668 ± 0.020 |
| GBDT-DKT+              | 0.793 ± 0.017 | 0.893 ± 0.007 | 0.357 ± 0.011 | 1.536 ± 0.017 |
| GBDT-DKT&SP            | 0.653 ± 0.047 | 0.816 ± 0.023 | 0.385 ± 0.007 | 1.431 ± 0.030 |
| GBDT-DKT+&SP           | 1.000 ± 0.000 | 1.000 ± 0.000 | 0.039 ± 0.007 | 1.961 ± 0.007 |
| LDA-SP                 | 0.369 ± 0.047 | 0.642 ± 0.037 | 0.425 ± 0.006 | 1.217 ± 0.042 |
| LDA-DKT                | 0.433 ± 0.021 | 0.675 ± 0.017 | 0.415 ± 0.003 | 1.260 ± 0.019 |
| LDA-DKT+               | 0.694 ± 0.036 | 0.838 ± 0.022 | 0.357 ± 0.010 | 1.481 ± 0.030 |
| LDA-DKT&SP             | 0.455 ± 0.037 | 0.700 ± 0.023 | 0.411 ± 0.007 | 1.290 ± 0.030 |
| LDA-DKT+&SP            | 0.452 ± 0.042 | 0.682 ± 0.015 | 0.412 ± 0.005 | 1.270 ± 0.020 |
| LR-SP                  | 0.369 ± 0.047 | 0.642 ± 0.037 | 0.424 ± 0.006 | 1.218 ± 0.042 |
| LR-DKT                 | 0.422 ± 0.029 | 0.671 ± 0.008 | 0.416 ± 0.002 | 1.255 ± 0.010 |
| LR-DKT+                | 0.695 ± 0.046 | 0.844 ± 0.026 | 0.356 ± 0.013 | 1.488 ± 0.038 |
| LR-DKT&SP              | 0.493 ± 0.032 | 0.732 ± 0.019 | 0.403 ± 0.006 | 1.329 ± 0.024 |
| LR-DKT+&SP             | 0.721 ± 0.042 | 0.857 ± 0.018 | 0.347 ± 0.013 | 1.510 ± 0.031 |
| (b) Test result        |         |        |        |              |
| GBDT-SP                | 0.301 ± 0.080 | 0.532 ± 0.103 | 0.436 ± 0.009 | 1.096 ± 0.108 |
| GBDT-DKT               | 0.321 ± 0.106 | 0.551 ± 0.138 | 0.441 ± 0.023 | 1.110 ± 0.159 |
| GBDT-DKT+              | 0.438 ± 0.144 | 0.637 ± 0.109 | 0.422 ± 0.029 | 1.216 ± 0.134 |
| GBDT-DKT&SP            | 0.383 ± 0.109 | 0.605 ± 0.050 | 0.425 ± 0.017 | 1.180 ± 0.062 |
| GBDT-DKT+&SP           | 0.455 ± 0.142 | 0.668 ± 0.074 | 0.434 ± 0.047 | 1.235 ± 0.117 |
| LDA-SP                 | 0.392 ± 0.176 | 0.641 ± 0.144 | 0.427 ± 0.028 | 1.214 ± 0.171 |
| LDA-DKT                | 0.426 ± 0.046 | 0.650 ± 0.066 | 0.420 ± 0.014 | 1.229 ± 0.078 |
| LDA-DKT+               | 0.406 ± 0.123 | 0.608 ± 0.109 | 0.460 ± 0.039 | 1.148 ± 0.148 |
| LDA-DKT&SP             | 0.448 ± 0.139 | 0.685 ± 0.089 | 0.418 ± 0.028 | 1.267 ± 0.118 |
| LDA-DKT+&SP            | 0.403 ± 0.173 | 0.629 ± 0.123 | 0.426 ± 0.028 | 1.203 ± 0.148 |
| LR-SP                  | 0.392 ± 0.176 | 0.641 ± 0.144 | 0.426 ± 0.025 | 1.215 ± 0.168 |
| LR-DKT                 | 0.382 ± 0.056 | 0.628 ± 0.053 | 0.427 ± 0.008 | 1.201 ± 0.060 |
| LR-DKT+                | 0.374 ± 0.125 | 0.605 ± 0.090 | 0.464 ± 0.048 | 1.142 ± 0.135 |
| LR-DKT&SP              | **0.458 ± 0.132** | **0.694 ± 0.089** | **0.414 ± 0.023** | **1.280 ± 0.107** |
| LR-DKT+&SP             | 0.390 ± 0.142 | 0.641 ± 0.080 | 0.457 ± 0.044 | 1.184 ± 0.124 |

The average and the standard deviation obtained in 5-fold cross-validation are reported. SVM-based models are not included because the importance of each feature cannot be determined in SVM model.
Table 7  The hyperparameter settings of each model reported in Table 6

| Model         | Params                          |
|---------------|---------------------------------|
| GBDT-SP       | max_depth: 3, min_samples_leaf: 5, n_estimators: 10 |
| GBDT-DKT      | max_depth: 3, min_samples_leaf: 1, n_estimators: 25 |
| GBDT-DKT+     | max_depth: 2, min_samples_leaf: 2, n_estimators: 50 |
| GBDT-DKT&SP   | max_depth: 3, min_samples_leaf: 1, n_estimators: 25 |
| GBDT-DKT+&SP  | max_depth: 5, min_samples_leaf: 10, n_estimators: 120 |
| LDA-SP        | solver: SVD                      |
| LDA-DKT       | solver: SVD                      |
| LDA-DKT+      | solver: SVD                      |
| LDA-DKT&SP    | solver: SVD                      |
| LDA-DKT+&SP   | solver: SVD                      |
| LR-SP         | C: 100.0, penalty: L1           |
| LR-DKT        | C: 100.0, penalty: L1           |
| LR-DKT+       | C: 100.0, penalty: L1           |
| LR-DKT&SP     | C: 100.0, penalty: L1           |
| LR-DKT+&SP    | C: 100.0, penalty: L1           |

Table 8  The mapping between the names of available skills in the ADM 2017 dataset and their IDs in this paper

| Skill Name                                      | ID          |
|------------------------------------------------|-------------|
| adding-decimals                                | 0           |
| addition                                       | 1           |
| algebra symbolization                          | 2           |
| algebraic-manipulation                         | 3           |
| application: compare expressions               | 4           |
| application: compare points                    | 5           |
| application: find slope in graph               | 6           |
| application: finding percentage of a number    | 7           |
| application: isosceles triangle                 | 8           |
| application: multi-column addition             | 9           |
| application: multi-column subtraction           | 10          |
| application: order of operations               | 11          |
| application: read points                       | 12          |
| application: simple multiplication              | 13          |
| area                                           | 14          |
| area-concept                                    | 15          |
| area-of-circle                                  | 16          |
| circle graph                                    | 17          |
| circumference                                   | 18          |
| combinatorics                                   | 19          |
| comparing-fractions                            | 20          |
| congruence                                      | 21          |
| discount                                        | 22          |
| divide-decimals                                 | 23          |
| divisibility                                    | 24          |
| division                                        | 25          |
| equation-concept                                | 26          |
| equation-solving                               | 27          |
| equivalent-fractions-decimals-percent           | 28          |
| evaluating-functions                            | 29          |
| exponents                                       | 30          |
| finding-percents                                | 31          |
| finding-percents-decimals-percent               | 32          |
| fraction-decimals-percents                      | 33          |
| fraction-division                               | 34          |
| fraction-multiplication                         | 35          |
| fractions                                       | 36          |
| g-geometry                                      | 37          |
| graph interpretation                            | 38          |
| inducing-functions                              | 39          |
| inequalities                                    | 40          |
| inequality-solving------------------------------ | 41          |
| integers                                        | 42          |
| interpreting-linear-equations                   | 43          |
| interpreting-numberline                         | 44          |
| isosceles-triangle                              | 45          |
| least-common-multiple                           | 46          |
| linear-area-volume-conversion                   | 47          |
| m-measurement                                    | 48          |
| making-sense-of-expressions-and-equations       | 49          |
| mean                                            | 50          |
| meaning-of-pl                                   | 51          |
| measurement                                     | 52          |
| median                                          | 53          |
| mode                                            | 54          |
| multiplication                                  | 55          |
| multiplying-decimals                            | 56          |
| multiplying-positive-negative-numbers          | 57          |
| n-number-sense-operations                       | 58          |
| noskill                                         | 59          |
| number-line                                     | 60          |
| of-means-multiply                              | 61          |
| order-of-operations                             | 62          |
| ordering-numbers                                | 63          |
| p-patterns-relations-algebra                    | 64          |
| pattern-finding                                 | 65          |
| percent-of                                      | 66          |
| percents                                        | 67          |
| perimeter                                       | 68          |
| point-plotting                                  | 69          |
| prime-number                                    | 70          |
| probability                                     | 71          |
| properties-of-geometric-figures                 | 72          |
| properties-of-solids                            | 73          |
| proportion                                      | 74          |
| pythagorean-theorem                             | 75          |
| rate                                            | 76          |
| rate-with-distance-and-time                     | 77          |
| reading-graph                                   | 78          |
| reciprocal                                      | 79          |
| reduce-fraction                                 | 80          |
| rounding                                        | 81          |
| scientific-notation                             | 82          |
| similar-triangles                               | 83          |
| simple-calculation                              | 84          |
| slope                                           | 85          |
| square-root                                     | 86          |
| statistics                                      | 87          |
| stem-and-leaf-plot                              | 88          |
| substitution                                    | 89          |
| subtracting-decimals                             | 90          |
| subtraction                                      | 91          |
| sum-of-interior-angles-more-than-3-sides        | 92          |
| sum-of-interior-angles-triangle                 | 93          |
| supplementary-angles                            | 94          |
| surface-area-and-volume                         | 95          |
| symbolization-articulation                      | 96          |
| transformations-rotations                       | 97          |
| transversals                                    | 98          |
| triangle-inequality                              | 99          |
| unit-conversion                                 | 100         |
| venn-diagram                                    | 101         |
Table 9 A summary of features used in each model after applying the RFE process

| Model          | Features                                                                 |
|----------------|--------------------------------------------------------------------------|
| GBDT-SP        | “AveCarelessness” and “AveResOfftask”                                    |
| GBDT-DKT       | knowledge state of 3 mathematical skills                                 |
| GBDT-DKT+      | knowledge state of 22 mathematical skills                                |
| GBDT-DKT&SP    | knowledge state of 6 mathematical skills and “AveKnow”                   |
| GBDT-DKT+&SP   | knowledge state of 55 mathematical skills, “AveKnow”, “AveCarelessness”,  |
|                | “NumActions”, “AveResEngcon”, “AveResConf”, “AveResFrust”, “AveResOff-    |
|                | task”, “AveResGaming” and “AveCorrect”                                    |
| LDA-SP         | “AveCarelessness”                                                        |
| LDA-DKT        | knowledge state of 7 mathematical skills                                 |
| LDA-DKT+       | knowledge state of 88 mathematical skills                                |
| LDA-DKT&SP     | knowledge state of 3 mathematical skills and “AveCarelessness”           |
| LDA-DKT+&SP    | knowledge state of 15 mathematical skills                                |
| LR-SP          | “AveCarelessness”                                                        |
| LR-DKT         | knowledge state of 7 mathematical skills                                 |
| LR-DKT+        | knowledge state of 28 mathematical skills                                |
| LR-DKT&SP      | knowledge state of 9 mathematical skills, “AveKnow”, “AveCorrect” and “Ave-|
|                | Carelessness”                                                            |
| LR-DKT+&SP     | knowledge state of 68 mathematical skills, “AveKnow”, “AveCarelessness”,  |
|                | “NumActions”, “AveResBored”, “AveResEngcon”, “AveResConf”, “AveResFrust”  |
|                | “AveCorrect”                                                             |

Effectiveness of Knowledge State

We have advocated the use of DKT or DKT+ in predicting whether a student will pursue a STEM field in his first job. In addition to the sample t-test conducted over the student profile previously, a further analysis of the relationship between the student’s DKT/DKT+ expected knowledge state and the STEM label is conducted.

DKT/DKT+ Knowledge State vs. Student Profile

In order to see whether the knowledge state is indeed helpful in predicting the STEM label, we project the student profile $x_{SP}$, the knowledge state $x_{KT}$ extracted in the DKT and the DKT+ models into a 1-dimensional space using LDA. The LDA projection is determined in a supervised manner, that is, the sample is projected according to the STEM label. If a feature set is capable of differentiating the STEM and the non-STEM classes, it is expected that the projected values between the two classes will differ significantly. Accordingly, after projecting all the samples into a 1-dimensional space, we can evaluate the distinguishing power of the feature set by visualizing the
projected values with a histogram. A feature set with a higher distinguishing power will result in two separate distributions. We also report the Fisher criterion to quantify the separation between these two distributions. The larger the value of Fisher criterion is, the more separate the two distributions are. The results are plotted in Fig. 2a, b and c.

As we can see, majority of the projections of the student profile between STEM and non-STEM classes overlap, with Fisher criterion of 0.27, indicating that the student profile alone may not be sufficient to differentiate the two classes. On the other hand, using the knowledge state obtained in the DKT or DKT+ models results in a better separation between STEM and non-STEM classes, with Fisher criterion of 0.62 and 0.59, respectively, implying that the student’s knowledge state in mathematics provides a rudimentary differentiation on whether a student is likely to pursue a career in the STEM field.

**DKT vs. DKT+: Their Discrimination Power on the STEM Label**

As observed from Fig. 2b and c, they both give a comparable distinguishing power between the STEM and the non-STEM classes. Although the Fisher criterion of DKT knowledge state is higher than that of DKT+ knowledge state, these distributions of STEM and non-STEM classes are very similar. To testify whether DKT model is better than DKT+ model, independent sample t-tests are conducted over each mathematical skill, similar to the one conducted in section Analyzing Student Profile. The t-scores of each skill in the t-tests are visualized by a heatmap (see Fig. 3). A negative value of t-score represents that the average knowledge state in the STEM class is higher than that in the non-STEM class, while a positive value of t-score represents the contrary. Moreover, the magnitude of the t-score indicates the confidence level of the difference of mean values between the two classes.

For majority of the skills, the independent sample t-tests show that the knowledge state extracted from the DKT+ has a better discrimination power between the STEM and the non-STEM classes, as their magnitudes of t-score are higher than those in the DKT. As for the skills with ID from 4 to 13 in the DKT model, most of them have a positive t-score with high magnitude. However, these skills are actually never addressed by any of the 467 labeled students in the training set. Hence, there is no ground truth as to whether the STEM class would have higher mean values in these skills than the non-STEM class. Yet, all of these skills are related to the application of some mathematical concepts that have been learned by students, so it would be expected that students in the STEM class show a higher competency in those skills. In other words, this implies that DKT+ is more reliable in estimating the student’s knowledge state.

---

6Refer to Table 8 for the mapping between ID and skill.
Adapt DKT+ Knowledge State as Normalized Learning Gain

Normalized learning gain (NLG) (Hake 1998) is used to measure how much a student learned after doing a learning activity, and it is calculated as:

$$\text{NLG} = \frac{\text{post-score} - \text{pre-score}}{1 - \text{pre-score}}$$

where the post-score represents the “test” score obtained by the student after performing the activity, while the pre-score represents the “test” score obtained prior to the activity. Thus, the larger the NLG, the more the student learned in the activity. To see whether the STEM class has a higher NLG than the non-STEM class, we adapt the NLG with the setting of a knowledge state extracted from the DKT+ model by treating the averaged knowledge state at each time-step to be the score of an activity. The averaged knowledge state at the first time-step can therefore be considered as the pre-score, while the averaged knowledge state at the last time-step can be treated as the post-score. Yet, considering that noise may exist in the knowledge state estimation, we use the average value of the knowledge state in the first ten time-steps to be the pre-score, and that of the last ten time-steps to be the post-score.

Fig. 3 The result of independent sample t-tests for each mathematical skill between DKT and DKT+ are visualized by a heatmap. The index in the horizontal dimension corresponds to the skill ID.
Then, we calculate the average NLG for both the STEM and the non-STEM classes. The average NLG for STEM students, $\bar{x}_{STEM}$, is 0.065 with a standard deviation of 0.172, while the average NLG for non-STEM students, $\bar{x}_{non-STEM}$, is 0.023 with a standard deviation of 0.146. We then perform a one-tailed hypothesis test to examine whether the average NLG in the STEM class is higher than that in the non-STEM class by formulating

\[
H_0 : \mu_{\bar{x}_{STEM}} - \mu_{\bar{x}_{non-STEM}} = 0 \\
H_1 : \mu_{\bar{x}_{STEM}} - \mu_{\bar{x}_{non-STEM}} > 0
\]

where $\mu$ represents the population mean. As a result, the null hypothesis $H_0$ is rejected with a p-value of 0.0072. It is concluded that STEM students have a higher knowledge gain in mathematics than non-STEM students after the use of the ASSISTments system, indicating the STEM class has a higher ability in learning mathematics. Accordingly, if a teacher would like to build up students’ interest in STEM during the middle school years, not only can she increase opportunities for students’ exposure to STEM activities, but also build up their ability in mathematics.

**Future Work**

A STEM predictor can help teachers identify potential STEM candidates so that teachers can organize STEM-related activities to strengthen their STEM ability. On the other hand, teachers can hold activities that are less STEM-intensive for non-STEM candidates in order to cultivate their interest in STEM. Therefore, improvement of STEM predictor can assist teachers to group students more precisely and better allocate resources. There are a few directions that can further improve the STEM predictor. First, more content in the clickstream can be exploited. In our experiments, only the content regarding the “skill” and “correct” is exploited in the clickstream record to extract the students’ knowledge state, while many other clickstream attributes, such as the affective states and the disengaged behaviors, are not used. They are actually a rich source to describe the student learning trajectory and potentially useful to discover some latent relationships between students’ learning trajectories and their career choices. Hence, unsupervised feature embedding methods could be applied to the clickstream data to obtain a condensed representation of a student learning trajectory. For example, one way would be to use a variational auto-encoder to encode the clickstream record to an embedded feature vector (Klingler et al. 2017).

Moreover, the training process can be merged from two-fold to one-fold, that is, end-to-end training. In our experiment, the training process is two-fold, where the DKT model is first trained to extract students’ knowledge state, and then a machine learning model is trained with the combined feature set. Accordingly, the machine learning model cannot directly learn the relationship between the clickstream data and the output label. An end-to-end learning model is therefore desirable because it
can automatically learn the latent representation of the necessary processing steps to map students’ clickstream data to their career choices.

Apart from possible extension for model improvement, another possible future direction is that how could we explain the STEM prediction and discover potential reasons why students pursue a STEM career. In this paper, we provide evidence to the correlation between the ability in mathematics and the likelihood of entering a STEM field, but we didn’t address the question of what kind of factors drive those STEM students to enter a STEM field. Are students who have entered a STEM field has an innate ability in Mathematics, or they studied Mathematics diligently so that they could enter a STEM field? If the reason why they enter a STEM field is not attributed to their innate ability, what is the reason behind that forms their desire? Knowing the reason why they are interested in STEM would help encourage students in studying STEM.

Conclusion

In this paper, we propose a two-fold training process incorporating the knowledge state extracted from the DKT or DKT+ model with the student profile to learn a STEM/non-STEM job classifier. Our experiments show that models trained with the combined features, that is, the student’s expected knowledge and the student profile, generally perform better, and analyses reveal that the expected knowledge state extracted from the DKT+ model is more robust than that from the DKT model. Moreover, a detailed analysis on the student’s knowledge state reveals that, when compared with non-STEM students, STEM students generally show a higher mastery level and a higher learning gain in mathematics. This provides evidence to the correlation between the ability in mathematics and the likelihood of entering a STEM field.

Acknowledgments

We thank the ASSISTments team for making the dataset publicly available for research purposes. The research has been supported by the project ITS/227/17FP from the Innovation and Technology Fund of Hong Kong.

References

Ashenafi, M.M., Riccardi, G., Ronchetti, M. (2015). Predicting students’ final exam scores from their course activities. In: Proceedings of the 2015 IEEE frontiers in education conference.

Baker, R.S. (2007). Modeling and understanding students’ off-task behavior in intelligent tutoring systems. In: Proceedings of the 25th SIGCHI conference on human factors in computing systems (pp. 1059–1068). ACM.

Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z. (2004). Off-task behavior in the cognitive tutor classroom: when students game the system. In: Proceedings of the 22nd SIGCHI conference on human factors in computing systems (pp. 383–390). ACM.

Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P. (2002). SMOTE: synthetic minority oversampling technique. Journal of Artificial Intelligence Research, 16, 321–357.

Clements, M. (1982). Careless errors made by sixth-grade children on written mathematical tasks. Journal for Research in Mathematics Education, 136–144.

Corbett, A.T., & Anderson, J.R. (1995). Knowledge tracing: modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction, 4(4), 253–278.
Davis, J., & Goadrich, M. (2006). The relationship between precision-recall and ROC curves. In: Proceedings of the 23rd international conference on machine learning, pp 233–240.

Fei, M., & Yeung, D.Y. (2015). Temporal models for predicting student dropout in massive open online courses. In: Proceedings of the 2015 IEEE international conference on data mining workshop, pp 256–263.

Geigle, C., & Zhai, C. (2017). Modeling MOOC student behavior with two-layer hidden Markov models. In: Proceedings of the 4th ACM conference on learning @ scale (pp. 205–208). ACM.

Hake, R.R. (1998). Interactive-engagement versus traditional methods: A six-thousandstudent survey of mechanics test data for introductory Physics courses. American journal of Physics, 66(1), 64–74.

He, H., Bai, Y., Garcia, E.A., Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: Proceedings of the 2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence), pp 1322–1328.

Khajah, M., Lindsey, R.V., Mozer, M.C. (2016). How deep is knowledge tracing. In: Proceedings of the 9th international conference on educational data mining, pp. 94–101.

Klingler, S., Wampfler, R., Käser, T., Solenthaler, B., Gross, M. (2017). Efficient feature embeddings for student classification with variational auto-encoders. In: Proceedings of the 10th international conference on educational data mining, pp. 72–79.

Lipton, Z.C., Berkowitz, J., Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning. arXiv:1506.00019.

Mi, F., & Fallings, B. (2017). Adaptive sequential recommendation for discussion forums on MOOCs using context trees. In: Proceedings of the 10th international conference on educational data mining, pp. 24–31.

Ocumpaugh, J., Baker, R., Gowda, S., Heffernan, N., Heffernan, C. (2014). Population validity for educational data mining models: A case study in affect detection. British Journal of Educational Technology, 45(3), 487–501.

Olah, C. (2015). Understanding LSTM networks. Colah. github. io. (Retrieved December 10, 2017 from http://colah.github.io/posts/2015-08-Understanding-LSTMs/).

Pavlik, P.I., Cen, H., Koedinger, K.R. (2009). Performance factors analysis – a new alternative to knowledge tracing. In Proceedings of the 14th international conference on artificial intelligence in education (pp. 531–538). Amsterdam: IOS Press.

Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L.J., Sohl-Dickstein, J. (2015). Deep knowledge tracing. In: Advances in neural information processing systems, pp. 505–513.

Reddy, S., Labutov, I., Joachims, T. (2016). Learning student and content embeddings for personalized lesson sequence recommendation. In Proceedings of the 3rd ACM conference on learning @ scale (pp. 93–96). New York: ACM.

Ren, Z., Ning, X., Rangwala, H. (2017). Grade prediction with temporal course-wise influence. In: Proceedings of the 10th international conference on educational data mining, pp. 48–55.

San Pedro, M.O., Ocumpaugh, J., Baker, R.S., Heffernan, N.T. (2014). Predicting STEM and non-STEM college major enrollment from middle school interaction with mathematics educational software. In: Proceedings of the 7th international conference on educational data mining, pp. 276–279.

San Pedro, M.O.C.Z., d Baker, R.S., Rodrigo, M.M.T. (2011). Detecting carelessness through contextual estimation of slip probabilities among students using an intelligent tutor for mathematics. In: Proceedings of 15th international conference on artificial intelligence in education, pp. 304–311.

Stewart, A., Bosch, N., D’Mello, S.K. (2017). Generalizability of face-based mind wandering detection across task contexts. In: Proceedings of the 10th international conference on educational data mining, pp. 88–95.

Yang, D., Sinha, T., Adamson, D., Rosé, C.P. (2013). Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In: Proceedings of the 2013 NIPS data-driven education workshop, vol. 11, pp. 14.

Yeung, C.K., & Yeung, D.Y. (2018). Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In: Proceedings of the 5th ACM conference on learning @ scale (pp. 5:1–5:10). ACM.
Zhang, J., Shi, X., King, I., Yeung, D.Y. (2017). Dynamic key-value memory networks for knowledge tracing. In: Proceedings of the 26th international conference on world wide web, pp. 765–774.
Zhou, G., Wang, J., Lynch, C.F., Chi, M. (2017). Towards closing the loop: Bridging machine-induced pedagogical policies to learning theories. In: Proceedings of the 10th international conference on educational data mining, pp. 24–31.

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

Chun-Kit Yeung¹ · Dit-Yan Yeung²

Dit-Yan Yeung
dyeyeung@cse.ust.hk
https://www.cse.ust.hk/faculty/dyeyeung/

¹ Find Solution Ai Limited, Sai Ying Pun, Hong Kong
² Hong Kong University of Science and Technology, Kowloon, Hong Kong