Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Automatic Control Laboratory

Report

Adaptive e-learning for educational recommendation system

Yu Tang
September 7, 2024

Advisors
Dr. Alberto Padoan
Dr. Mathias Hudoba de Badyn
Scott Sussex
Prof. Dr. John Lygeros

Keywords: ‘Intelligent tutoring system, Recommendation system, LonCAPA dataset, Collaborative filtering, Knowledge tracing

1 Abstract

Massive data and the increasing demand of personalized learning motivates the design of Intelligent tutoring system (ITS). To achieve this goal, two key components are: knowledge state estimator and education recommender. A knowledge state estimator is required to estimate learners' knowledge state and predict future performance. Thereby, an education recommender could be built based upon knowledge estimation or performance prediction. In this work, we investigate different approaches to building an knowledge state estimator (predictor). Specifically, we discuss Collaborative filtering based methods and knowledge tracing based methods. Moreover, we have a novel dataset called LonCAPA, and we show the feasibility of building an knowledge state estimator and performance predictor from the LonCAPA dataset.

Contents

List of Figures

List of Tables

2 Introduction

With the increasing amount of information and the arising complexity of content, the information search and gathering task becomes time-consuming for users. Recommendation system can be leveraged to cope with the information overload by means of providing personalized recommendations to individuals based on their preferences, behavior, and past interactions with a system or platform. A wide range of its applications include e-commerce, social media, e-business, e-learning, e-resource services, etc [35]. Specifically, in the realm of educational technology, Educational data mining (EDM) draws attention of lots of researchers as it utilizes statistical, machine-learning, and data-mining algorithms over various educational data, and exploits computational approaches in order to study educational questions [45]. Under the big realm of EDM, Intelligent tutoring systems (ITSs) is an emerging research area, which focuses on enhancing the learning experience of each particular student [3]. It can utilize and produce extensive dataset, and thereby provide personalized feedback and improve students' learning process. A critical problem in the field of ITSs is effectively and accurately modeling the performance of students' interaction with an ITS. In particular, given the past data of a students' learning history, an ITS model should be able to observe the current state of a studentâs knowledge and, based on that, predict his/her future performance [19]. A competent learner performance model would then lay a solid foundation for constructing an educational recommendation system, which accomplishes the goal of ITSs - offering personalized support to individual learner. To build a educational recommendation system, either heuristic policies [13, 22, 29, 42] or deep reinforcement learning based policies [2] exist.

Two key components that contribute significantly to the efficacy of learner performance model of ITSs are Collaborative filtering recommender approaches and Knowledge tracing. Collaborative filtering based methods are generally suitable for datasets where no temporal and dynamic information is included. Meanwhile, each student answers similar number of questions only once. If a questions is answered multiple times by a single student, only the first attempt is considered by the model. Datasets like mid-term exam would possess such characteristics, and thus Collaborative filtering methods can be leveraged to estimate students' ability and provide prompt feedback for better preparation of the finals. On the other hand, knowledge tracing models can leverage temporal information encoded in the dataset. In the meantime, each student may have different and long learning trajectories. And each question/exercise might appear multiple times; some non-feedback learning materials (e.g., video, textbook pages) might also be recorded in the dataset. Such scenario could exist in online learning or homework tutorials in a course. In this article, we will review and benchmark both Collaborative filtering algorithms and Knowledge Tracing algorithms. Some open datasets as well as our synthetic dataset and private LonCAPA dataset would be leveraged to evaluate the performance of algorithms. This article is structured as follows. We review and provide some preliminaries regarding Collaborative filtering in Section 3. We present various categories of Knowledge tracing algorithms in Section 4. The used datasets are illustrated in Section 5. The experiment results of both Collaborative filtering based algorithms and knowledge tracing algorithms are presented in Section 6.

3 Collaborative filtering

Collaborative filtering is a technique commonly used in recommendation systems to provide personalized advice to users. No matter user-based collaborative filtering [44] or item-based collaborative filtering [47], the fundamental intuition behind collaborative filtering is to match the users' preferences and behavior with other similar users and items. Essentially, it utilizes the collective insight of a group of users and the similarities among items to give predictions or recommendations for an individual user [33, 43, 47].

Matrix Factorization (MF), as one type of model-based collaborative filtering, is a commonly investigated technique and plays a crucial role in the recommendation systems. In the context of ITSs, the matrix represents interactions between learners and educational resources. This technique decomposes this question-learner interaction matrix into low-dimensional latent space. The idea is to learn a compact representation of the original question-learner interaction matrix, capturing hidden patterns and preferences in the dataset. Thereby, the recommender system can make predictions in case of missing values in the original matrix, give each individual a personalized ranking over a set of questions/knowledge concepts, and recommend appropriate exercises or learning materials that align with their individual needs and learning style. MF becomes a popular technique in that it can efficiently and flexibly handle issues such as matrix sparsity and scalability [10, 26]. Multiple algorithms of MF exist in current literature, including Singular Value Decomposition (SVD) [21, 25, 26], Alternating Least Squares (ALS) [20, 54], Non-negative Matrix Factorization (NMF) [30, 32, 37]. In this report, we employ one most popular matrix factorization technique - NMF.

3.1 Non-negative matrix factorization

Non-negative matrix factorization (NMF) is a technique to factorize a given matrix V into two matrices W and H, with the constraint that all three matrices have no negative entries [48]. In other words, it approximates the matrix V by finding W and H whose multiplication minimizes the error function. The problem formulation is defined by Eq. 1

$$\begin{aligned} \underset{W,H}{\text{minimize}} \quad & \|\boldsymbol{V} - \boldsymbol{WH}\|_F \\ \text{subject to} \quad & W \geq 0, \quad H \geq 0 \end{aligned} \tag{1}$$

The reconstruction error of NMF is defined as $E = \|V - WH\|_F / \|V\|_F$. In our scenario, we process the data into a given question-response matrix V which has dimension $R^{Q \times N}$, where Q is the number of total questions and N is the number of students. Thereby, the W matrix is of size $W \in R^{Q \times K}$, and H matrix is of size $H \in R^{K \times N}$, where K is the number of estimated latent concepts, generally a hyper-parameter to be tuned in terms of reconstruction error. A physical interpretation is that NMF decomposes the given question-response matrix into a question-structure matrix and a student knowledge matrix, so that it achieves latent feature extraction. There are various algorithms to achieve NMF such as convex NMF [16], nonnegative rank factorization [8][9], convolution NMF [7], etc. Due to the simplicity of implementation, multiplicative update rule and coordinate descent method [31] are two widely used algorithms for NMF, and they can be easily implemented via open-source libraries like *nimfa*[1] and *sklearn*[2].

There is a uniqueness issue associated with NMF algorithms in general [49]. For any pair of solution (W, H), there always exists a diagonal matrix $D \geq 0$ such that $(WD)(D^{-1}H) = (WH)$, which makes the solution for (W,H) not unique. Yet, for different independent runs, even though the non-uniqueness issue yields totally different output of (W,H), the reconstruction error is almost the same. Further experiment shows that the variation of final reconstruction matrix in different runs, i.e., $\|W_1H_1 - W_2H_2\|_F / \|V\|_F$, is small, where the subscript number denotes the order of experiments. Thereby, since we observe that $W_1H_1 \approx W_2H_2$, we can find a transformation matrix T such that $W_1T \approx W_2$ and $T^{-1}H_1 \approx H_2$. More control over the non-uniqueness issue of NMF is achieved with sparsity constraints [17].

[1]https://nimfa.biolab.si/nimfa.methods.factorization.nmf.html

[2]https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.NMF.html

4 Knowledge tracing

Knowledge tracing (KT), also known as Latent knowledge estimation, is a data-driven approach used in educational technology that models a student's knowledge state and predicts a learner's future response based on his/her past learning sequence. It achieves this by measuring, implicitly or explicitly, what Knowledge components (KCs) a learner acquires at a specific time. Knowledge components (KCs) refer to the distinct skills, concepts, questions, and exercises that are recorded in the dataset [1, 19, 34]. Each question/exercise might require more than one skill/concept. It is important to distinguish between question/exercise and concept/skill in the data pre-processing steps before employing KT algorithms. Some dataset are embodied with both information of questions and concepts, while others just have either. It is common to convert each unique combination of KCs into a new KC . To be consistent and clear, we follow the similar definition as in [19], where each single question/exercise is termed as item, and each skill/concept/type is termed as knowledge concepts.

Knowledge tracing (KT) is formulated as: given observations of a studentâs historical interactions sequence $\mathbf{X_{1:t}} = \{x_1, x_2, ..., x_t\}$, predict whether the student answers a new interaction x_{t+1} correctly [14]. The input at a specific timestamp t takes the form of a tuple $x_t = (q_t, r_t)$, which contains the item being answered q_t and the corresponding result r_t. Therefore, KT task aims to find the probability of $P(r_{t+1} = 1|q_{t+1}, \mathbf{X_{1:t}})$. There are many existing KT techniques, and can be generally categorized into three tracks: Markov process probabilistic methods, factor analysis methods, and deep learning methods.

4.1 Markov process methods

Markov processes describes a sequence of events where the conditional probability of transitioning to any future state depends solely on the current state and time, in spite of the history [46]. Bayesian Knowledge Tracing (BKT) is the most widely-used probabilistic method, which assumes the learning process follows a Markov process [14]. Specifically, BKT employs a Hidden Markov Model (HMM), where the binary observed state is the result of student's answer $r_t \in \{correct, incorrect\}$, and the hidden state is a binary student's knowledge state on each KC: $\{learned, unlearned\}$. BKT estimates and traces the learnerâs knowledge state independently for each KC, and it trains four parameters per KC. Therefore, the total number of trainable parameters for a standard BKT is $4K$, where K is the number of KCs. Table 1 presents the definition of these four parameters, and the update process of standard BKT is illustrated in Figure 2.

Training parameters	Description
$P(L_0)$	Probability that the skill is known before doing any exercise
$P(T)$	Probability that the skill is learned after doing exercise
$P(S)$	Probability of making a slip in a learned state
$P(G)$	Probability of guessing exercise correctly in an unlearned state

Table 1: Description of four trainable parameters per knowledge component for a standard BKT model

The standard BKT assumes the skill is never forgotten once learned by a student. At each time step $n \geq 1$, the probability of a skill becoming a learned state at this interaction $P(L_n)$ is updated based on transition parameter $P(T)$ as shown in Eq. 2.

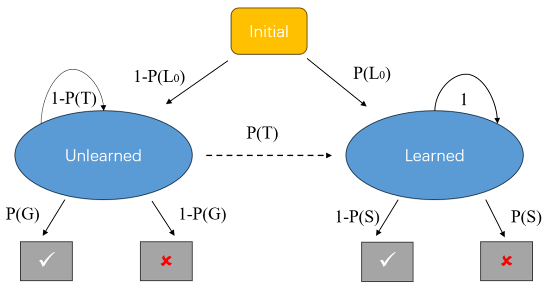


Figure 2: The update rule of standard BKT. *Unlearned* and *learned* are binary latent knowledge states, and binary observations are represented by "tick/cross" mark. *Initial* means the starting state; $P(L_0)$ represents the probability of the starting state being *learned.*

$$P(L_n) = P(L_{n-1}|x_n) + (1 - P(L_{n-1}|x_n)) * P(T) \tag{2}$$

,where $x_n \in correct, incorrect$ is the observation at n_{th} step, and the posterior probability $P(L_{n-1}|x_n)$ is calculated by Bayesian inference in Eq. 3.

$$\begin{aligned} P(L_{n-1}|x_n = correct) &= \frac{P(L_{n-1}) * (1 - P(S))}{P(L_{n-1}) * (1 - P(S)) + (1 - P(L_{n-1})) * P(G)} \\ P(L_{n-1}|x_n = incorrect) &= \frac{P(L_{n-1}) * P(S)}{P(L_{n-1}) * P(S) + (1 - P(L_{n-1})) * (1 - P(G))} \end{aligned} \tag{3}$$

And the estimation probability of getting correct in next interaction $P(C_{n+1})$ is calculated by the sum probability of both guessing correctly in an unlearned state and making no slip in a learned state:

$$P(C_{n+1}) = P(L_n) * (1 - P(S)) + (1 - P(L_n)) * P(G) \tag{4}$$

Noting that there are only four trainable parameters per skill, and the other parameters, $P(L_n)$ and $P(C_{n+1})$, are not trained but directly updated by Markov process. Some other BKT variants are proposed after the standard BKT, such as adding constraints of guessing and slipping estimates [5, 15], considering external help and non-feedback materials [6], including each exercise difficulty [39], modelling the connection and dependency of knowledge concepts to achieve dynamic BKT [23]. Meanwhile, since the standard BKT treats all students equally with the same prior knowledge capability and learning rate, it cannot be personalized effectively to each individuals. Thus, prior mastery knowledge $P(L_0)$ for each learners is individualized in [40], and learning rate $P(T)$ is student-specific in [52]. Work in [24] also individualizes the guessing $P(G)$ and slipping $P(S)$ parameters for each student.

4.2 Factor analysis methods

The key idea of Factor analysis method is to estimate student performance by fitting a function, usually a logistic function [1]. Therefore, it is also sometimes referred to as Logistic regression method. Different feature vectors in logistic regression model yield different methods, such as Item Response Theory (IRT) [18], Performance Factors Analysis (PFA) [41], DAS3H [12]. The most commonly employed method is It IRT, which is applied in many education assessment and measurement systems like Graduate Record Examination (GRE) Aptitude Test. The relationship between an individual's response to an item and their latent trait is typically modeled using a logistic function, such as the 1-parameter logistic (1PL) model or the 3-parameter logistic (3PL) model. Eq. 5 is the formulation of 1PL model.

$$P(\theta) = \frac{1}{1 + \exp(-1(\theta - b))} \tag{5}$$

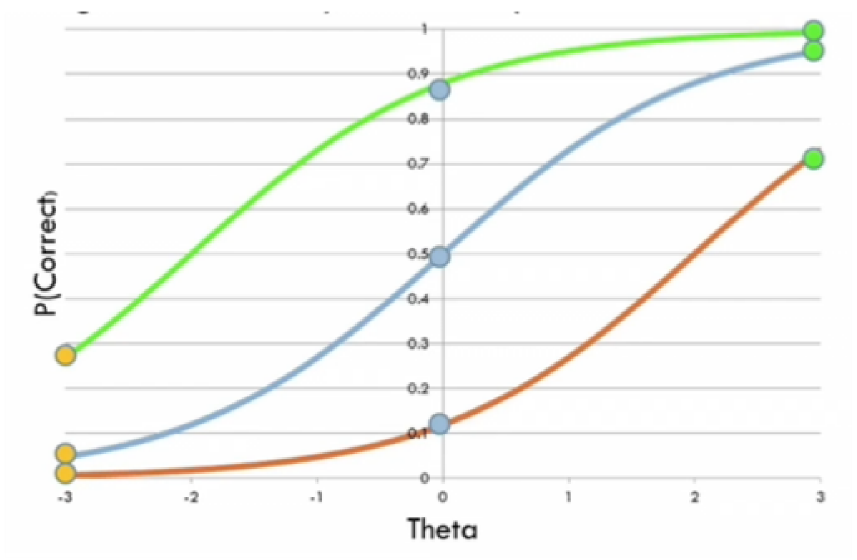


Figure 3: Illustration of 1-PL IRT model. Green line has difficulty parameter $b = -2$ and represents easy question. Blue line has $b = 0$ and represents medium-level question. Orange line has $b = 2$ and represents hard question.

where each learner has ability θ and each question has a difficulty level b. The probability of an individual correctly answering item is modelled by $P(\theta)$. If we incorporate a guessing parameter c and a discrimination parameter a, we obtain the 3-PL model as in Eq. 6. The guessing parameter represents the probability of guessing the correct response when the latent trait level is extremely low, and the discrimination parameter controls the slope of the logistic curve and thereby measures how well the item discriminates between individuals with different levels of the ability.

$$P_i(\theta) = c + \frac{1 - c}{1 + \exp(-a_i(\theta - b_i))} \tag{6}$$

Figure 3 demonstrates a simple example of 1-PL model. It plots the the probability $P(\theta)$ versus the ability parameter θ. There is always a positive relationship between θ and $P(\theta)$. Three different lines depict three different values of item difficulty b. The easier the question, intuitively the larger the probability $P(\theta)$. Green line has difficulty parameter $b = -2$ and represents easy question. Blue line has $b = 0$ and represents medium-level question. Orange line has $b = 2$ and represents hard question.

4.3 Deep learning methods

Deep learning method for knowledge tracing is firstly proposed in [42], which uses traditional Recurrent Neural Networks (RNNs) and its variant, Long Short Term Memory (LSTM), to represent complex features in a much higher dimensional and continuous space. Rather than using hand-crafted features, it enables flexible function approximation and the features are learned directly from data. This method has become a competitive alternatives to classical probabilistic methods or factor analysis methods by virtue of the advent of increasingly large scale datasets [19]. Figure 4 depicts the structure of the most original Deep knowledge tracing (DKT) method. It employs a RNN network to map an input sequence of vectors $x_1, \ldots, x_T$, to an output sequence of vectors $y_1, \ldots, y_T$. The input sequence is the student past interactions. A common practice to deal with inputs is to set each interaction point x_t to be a one-hot encoding vector of student interaction tuple $x_t = (q_t, a_t)$. Here, q_t represents which question is answered at timestamp t and a_t represents the corresponding result. And for the outputs, since the dataset labels are binary: $\{correct, incorrect\}$, it becomes a binary classification problem and the output is usually interpreted as the probability. Specifically, at each timestamp, y_t is a vector of length equal to the number of questions, where each entry represents the probability of getting the particular question correct. The inputs and outputs are related by a sequence of hidden states $h_1, \ldots, h_T$, which successively encodes relevant information from past trajectory. The network is defined by the following Eq. 7

$$\begin{aligned} h_t &= tanh(W_{hx}x_t + W_{hh}h_{t_1} + b_h), \\ y_t &= \sigma(W_{yh}h_t + b_y) \end{aligned} \tag{7}$$

where the *tanh* and *sigmoid* $\sigma(.)$ operations are applied element-wise. The trainable parameters in the model is input weight matrix W_hx, recurrent weight matrix W_hh, initial hidden state h_0, and readout weight matrix W_yh. The predicted probability of answering question q_{t+1} correctly can be read from the corresponding entry in y_t [42]. We can then compare it with the actual label a_{t+1} to calculate the loss. Binary classification problems usually use binary cross-entropy loss for network training [11]. Thus, the training objective function of each single student in this problem is set to be the summation of all binary cross entropy loss over the learning trajectory, and the objective is minimized using stochastic gradient descent methods and its variants.

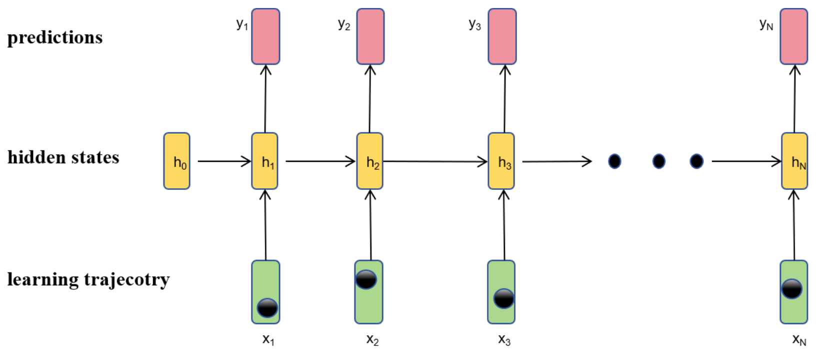


Figure 4: Depiction of the vanilla Deep Knowledge Tracing (DKT) model.

Since the proposal of the first DKT, there are many following variants of DKT-family. For example, DKT+ model adds regularization terms in the loss function to enhance the consistency in DKT prediction [51]. Forgetting behavior is considered by incorporating multiple forgetting information in the DKT-Forget model [36]. Because the original DKT possesses an issue that the hidden layers are not physically interpretable, Dynamic Key-Value Memory Network (DKVMN) is proposed for the hope of solving this issue [53]. It borrows the idea of Memory-Augmented Neural Networks and uses two matrices, one static and one dynamic, to stores a exercise-concept mapping and a student latent knowledge state. A synthesis of the item response theory (IRT) model and DKVMN model is proposed, called Deep-IRT, to provide more explainable interpretation of both students and exercises [50]. Furthermore, Self-attentive knowledge tracing (SAKT) employs the Transformer architecture to shift the structure from recurrent neural networks to attention mechanisms [38].

5 Datasets

In this section, we present an overview of datasets that are used in our experiment. Both open datasets and our novel LonCAPA dataset [27, 28] will be discussed. For collaborative filtering method, which is a fairly mature area, we do not benchmark on open datasets; yet we directly test it on our LonCAPA dataset. And for knowledge tracing methods, we will introduce and benchmark some open datasets commonly used in this field, as well as LonCAPA dataset.

5.1 LonCAPA dataset

LonCAPA dataset is obtained from an open-source learning content management and assessment system in college-level introductory physics courses [28]. It contains logs of 34 physics courses. The dataset contains a mixture of completely online and hybrid courses. For each individual course, the logs contain information of three modes: *Homework*, *Graded by TA*, *Exam. Homework* mode is an online learning setting with immediate feedback, thus containing temporal information. Each student has different and long learning trajectories. Each question might appear multiple times and some non-feedback materials also appear in the log. This characteristics of this mode makes it suitable for applying knowledge tracing approaches. On the other hand, *Exam* mode contains no temporal information. Each student answers similar number of questions only once. Therefore, collaborative filtering methods are suitable for this mode of dataset. Since *Graded by TA* mode has very few data points, it is not treated for experiment.

A total of 9349 individuals accessed 2464 different content pages and 3703 different online problems, leading to 28825817 access interactions in total. Some of the courses are recurring every year and therefore, a large portion of the questions are overlapped even though they are marked as different courses. When implementing knowledge tracing algorithms, we select some of the mostly overlapped courses ($1st$, $3rd$, $5th$, $19th$, $22nd$, $28th$), and combine them together into one big course, called *Lon-Comb*. The reason for this operation is to enlarge the training and testing dataset in order to match the size with other open datasets without significantly increasing the network complexity. We also select one single course, $4th$ course, as comparison and mark it as Lon-Course4. The reason for selecting it is because the statistics in this dataset is clean. There is no *ungraded* result. The numbers of students and unique questions in the course are comparably large in term of traditional classroom setting.

5.2 Knowledge tracing datasets

We leverage four open datasets and our own LonCAPA dataset in our implementation of knowledge tracing algorithms. For LonCAPA dataset, we select some portion of data to be used in experiment. Specifically, we select some of the mostly overlapped courses out of the 34 courses, and combine them together into one big course, called *Lon-Comb*. The reason for this operation is to enlarge the training and testing dataset in order to match the size with other open datasets without significantly increasing the network complexity. We also select one single course, $4th$ course, as comparison and mark it as *Lon-Course4*. The reason for selecting it is because the statistics in this dataset is clean. There is no *ungraded* result. The numbers of students and unique questions in the course are comparably large in term of traditional classroom setting. Table 2 summarizes the characteristic of the selected portion of LonCAPA dataset, as well as other open datasets.

ASSISTments2009[3]: The dataset is obtained from the ASSISTments online tutoring platform, containing 4151 students with 325637 interactions. The number of questions is 26688, out of 124 latent concepts. The mean sequence length (i.e., learning trajectory) of a student is 78. The

[3]https://sites.google.com/site/assistmentsdata/home/2009-2010-assistment-data/skill-builder-data-2009-2010

Dataset	No. questions	No. concepts	No. students	No. interactions	Sequence length
ASSISTments 2009	26684	124	4217	525534	124.6 ±307.6
ASSISTments 2015	N/A	100	19840	683801	34.5 ±41.4
Statics2011	1223	N/A	333	261947	786.6 ±534.8
Synthetic	50	5	2000	100000	50 ±0
Lon-Course4	227	N/A	860	409163	475.8 ±174.6
Lon-Comb	720	N/A	2078	1399954	673.7 ±223.5

Table 2: The Statistics of the knowledge tracing datasets

correct rate of this dataset is 65.84%.

ASSISTments2015[4]**:** This dataset only records student responses on 100 latent concepts. The information regarding the number of questions is not provided. After preprocessing, it has 683,801 effective interactions given from 19,840 students. It has a shorter mean sequence length per student, which is 34. Because it has much larger number of students compared to ASSISTments2009 dataset, the average number of interactions per skill and per student is smaller even though it records a large number of total interactions.

Statics2011[5]**:** This dataset is obtained from an college-level engineering statics course, with 189,297 interactions from 333 students and 1223 questions. The latent concepts tag is not available. The mean sequence length of a student's learning history is 568.

Synthetic[6]**:** This synthetic dataset is first created by [42]. Afterwards, many following literature employs this dataset for benchmarking purpose. It simulates 2000 virtual students taking in total 50 questions out of five latent concepts. The learning trajectory of each virtual student is the same as 50. Each question has its own difficulty β and each student has a knowledge state θ towards each of the five latent concepts. The probabilty of a student getting a question a correctly is modelled by classic IRT **??**: $p(a = 1|\theta, \beta) = c + \frac{1-c}{1+exp(-(\theta-\beta))}$, where c is the probability of a random guess, and set to be 0.25 in the dataset.

6 Experiments

We present performance results of both domains: Collaborative filtering and Knowledge tracing. In collaborative filtering realm, we apply NMF algorithms on different courses of LonCAPA. In knowledge tracing field, we benchmark on open datasets in 5.2 and our LonCAPA dataset using the following algorithms: BKT, IRT, DKT, DKVMN.

6.1 Collaborative filtering experiments

As discussed in Section 3.1, there is a latent hyper-parameter K, which can be interpreted as the estimated number of underlying concepts in an exam. The value of K also indicates the amount of original information to be reserved by matrix factorization. We try to vary K from a very small number to a number close to Q, the number of total questions in an exam. Not surprisingly, the reconstruction error E reduces as K increases. For example, Figure 5 shows a the change of reconstruction error versus K for LonCAPA 1*st* course. There are 31 questions

[4]https://sites.google.com/site/assistmentsdata/datasets/2015-assistments-skill-builder-data
[5]https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=507
[6]Synthetic-5:https://github.com/chrispiech/ DeepKnowledgeTracing/tree/master/data/synthetic

and 297 students in this exam. The result is obtained by averaging 10 independent experiments.

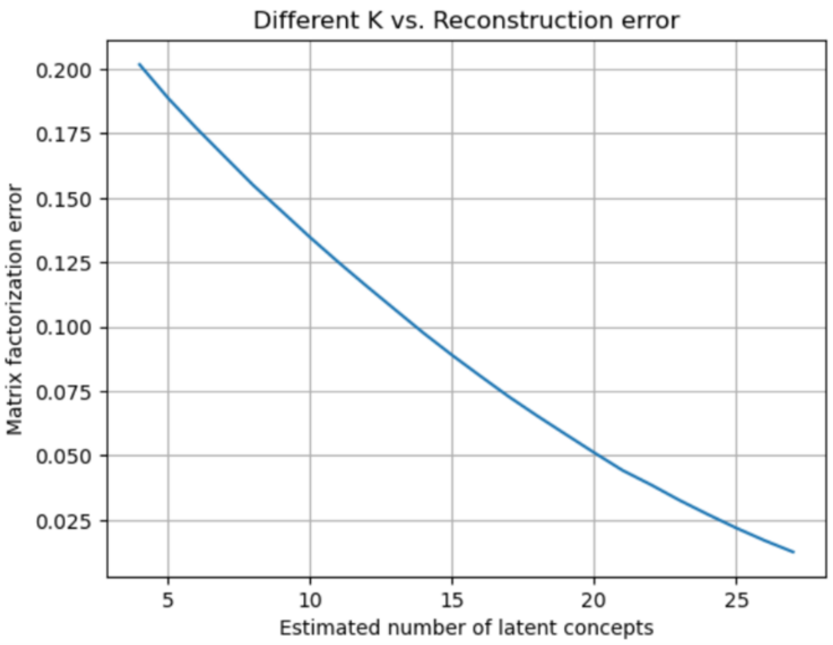


Figure 5: The amount of reconstruction error as a function of K, the estimated number of latent concepts for LonCAPA $1st$ course *exam* mode.

If K is given in the dataset, cross-validation method could be employed to perform hyper-parameter selection. However, this latent number K is sometimes not available in datasets, for example, our LonCAPA. Thereby, we could only tune it based on our requirement on reconstruction error and possibly some prior knowledge on the course structure. The obtained W matrix can interpreted as our estimation on the exam structure, and H matrix is our estimation of students' mastery level towards different latent concepts. For instance, if we select $K = 5$ for $1st$ course, one single column in H is as Table 3. Here, the label of the five latent concepts are based on our prior knowledge yet not directly available in LonCAPA dataset. The larger the value in each entry, the better the mastery level towards this concept. If we simply average across all columns, we can get the overall estimated student knowledge level towards this course, as indicated in Table 4.

One thing to note is that traditionally when utilizing matrix factorization technique, there are usually missing entries in the given matrix, and those missing values are eventually filled by estimation. In our LonCAPA dataset *exam* mode, after data pre-processing, we filter out outlier data and the remaining part are generally fully observed. Nevertheless, matrix factorization method could still be leveraged if missing entries exist. In other words, the results of non-response question could also be estimated, and hence the overall knowledge level can be evaluated even though some questions are not answered.

Newton first law	Torque	Friction	Ohm's law	Work and energy
0.546	0.131	0.550	0.260	0.525

Table 3: Estimated knowledge concept state of a single learner

Newton first law	Torque	Friction	Ohm's law	Work and energy
0.345	0.375	0.311	0.235	0.343

Table 4: Average estimated knowledge concept state of all learners

6.2 Knowledge tracing experiments

In the implementation of BKT, we leverage the package from [4], which fits a BKT model using Expectation Maximization[7]. In the implementation of factor analysis method, we employ the code from [19], which runs on DAS3H repository[8]. In the implementation of deep learning methods, since many literature report a comparable validation performance of all deep learning based KT methods [19, 34, 50], we run on some of the most commonly used algorithms to give a sanity check and benchmarking comparison, so as to show the effectiveness and feasibility of running state-of-the-art KT algorithms on our novel LonCAPA dataset. Specifically, we run BKT, IRT, vanilla DKT, and DKVMN algorithms, and compare them using Area Under Curve (AUC) score, which is a commonly used metric for KT tasks [36, 42, 50, 53]. For our LonCAPA dataset, We split the training set and test set by a portion of 4 : 1, as this is a common practice in other open datasets mentioned in Section 5.2. In each run, early stopping is performed to obtain the corresponding AUC score. The benckmarking result is shown in Table 5.

Model	ASSISTments 2009	ASSISTments 2015	Statics 2011	Synthetic	Lon-Course4	Lon-Comb
BKT	0.7595	0.7010	0.7500	0.8020	0.5854	0.7079
IRT	0.7985	0.7015	0.8476	0.6887	0.7145	0.7290
DKT	0.8423	0.7271	0.8440	0.8301	0.7262	0.7476
DKVMN	0.8002	0.7223	0.8317	0.8271	0.7153	0.7256

Table 5: Test AUC score for all datasets using different knowledge tracing models.

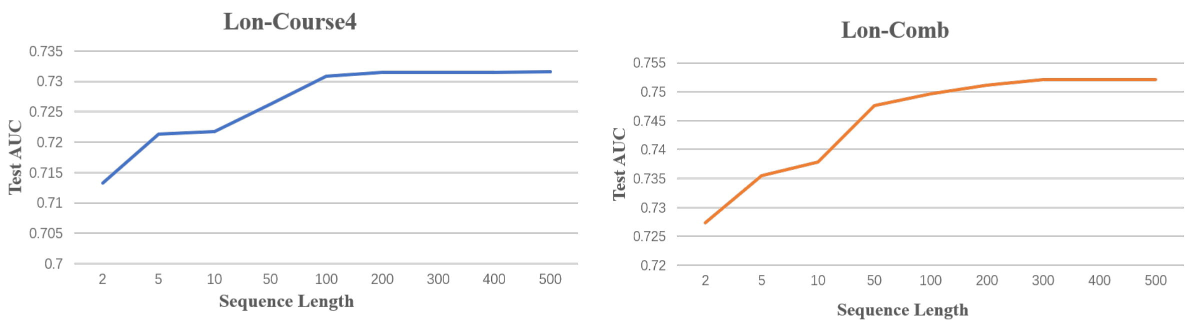
Sequence length and test performance

When running on deep learning based algorithms, there is an additional hyper-parameter called sequence length. Here, the definition of sequence length is borrowed from [53], which refers to the post-processed length of learning trajectory of each student. For example, if the original learning trajectory has a length of 420 and the sequence length is 50, then after data processing it becomes nine "virtual" students, each having 50 learning steps. We do paddings for any total sequence that is not divisible by sequence length and mask those padding out during training and testing. Note that changing the sequence length hyper-parameter may only have impact on DKT-family methods but not the BKT or Logistic regression methods. Because deep learning based KT, either RNN or Transformer, would incorporate the past inputs with current input for training and predicting. Thus, the sequence length represents how much of the past information to be encoded. For our LonCapa datasets, we run vanilla DKT model with different sequence lengths, specifically of size 2, 5, 10, 50, 100, 200, 300, 400, 500, and summarize the results in Table 6. We also plot these data in Figure 6 for a better visualization.

[7]https://github.com/CAHLR/pyBKT-examples
[8]https://github.com/BenoitChoffin/das3h

Model	Lon-Course4		Lon-Comb	
	s . len	test auc	s . len	test auc
DKT	2	0.7132	2	0.7273
	5	0.7213	5	0.7355
	10	0.7217	10	0.7397
	50	0.7262	50	0.7476
	100	0.7309	100	0.7497
	200	0.7315	200	0.7512
	300	0.7315	300	0.7521
	400	0.7317	400	0.7522
	500	0.7316	500	0.7522

Table 6: Test AUC score of LonCapa dataset as varying the input sequence length.



(a) Lon-Course4

(b) Lon-Comb

Figure 6: The test AUC scores vary as the input parameter sequence length changes. (a) plots the results for Lon-Course4 while (b) plots the results for Lon-Comb. The sequence length increases from a very small number to a large number comparable to the average total trajectory length. For every experiment, early stopping is performed.

We find that the performance of DKT with sequence length of 2 is similar to that of logistic regression model. Meanwhile, as the sequence length increases, the AUC score also gradually improves and levels off until the sequence length is approximately half of the average total trajectory length. It also gives an sanity check on the LonCAPA dataset itself that the LonCAPA historical information is helpful for future prediction, and hence this dataset is feasible for KT algorithms.

7 Conclusions

Intelligent tutoring systems (ITSs) requires a knowledge state estimator to design overall education recommendation system. The state estimator takes a learning history of past interactions as inputs, and then estimates the current state of a learnerâs knowledge and/or predicts his/her future performance. In this project, we employ a novel dataset LonCapa. We run both collaborative filtering algorithms and knowledge tracing algorithms on this dataset, and benchmark it with other open knowledge tracing datasets to validate the feasibility of using LonCAPA to construct a recommender system for education.

For collaborative filtering domain, we use LonCAPA *exam* mode due to the characteristics of this mode: It contains no temporal/dynamic information; each student answers similar number of questions only once; there might have missing entries. We leverage one most commonly used Collaborative filtering algorithm, NMF. Since the dataset does not record the number of knowledge concepts K, it has to be tuned based on reconstruction error and other prior information. In general, the larger the K, the lower the reconstruction error. The obtained W matrix can be interpreted as the exam structure matrix and the H matrix is the student knowledge state matrix, which could potentially be used for recommendation system design. The issue for using this method is that matrix factorization algorithms generally yield non-unique solution.

For knowledge tracing methods, we use LonCAPA *Homework* mode data because it is an online learning scenario containing temporal information. Each student has different and long learning trajectories, and each question might appear multiple times with some non-feedback materials being involved. We run all three categories of knowledge tracing methods: Markov process methods, Factor analysis methods, Deep learning methods. Factor analysis methods and deep learning methods outperform the traditional Markov process methods. We run deep learning methods with different sequence length on LonCAPA. The result shows that LonCAPA historical information is helpful for future prediction and hence LonCAPA dataset is suitable for knowledge tracing tasks. Thereby, we can use knowledge tracing models to build a student simulator for education recommendation system. One issue regarding using knowledge tracing methods, especially DKT, is that the hidden state is not physically explainable.

In terms of future work, we could quantitatively compare LonCAPA against other open datasets in order to help us better understand the performance of building a knowledge state estimator from a real-classroom dataset. Meanwhile, there are other ways of constructing student simulator such as using large language model. We could possibly compare our student simulator with that from large language model. Then based upon a satisfactory student simulator, we could possible learn a recommendation policy using deep Reinforcement learning techniques and evaluate it using the simulator.

Bibliography

[1] Ghodai Abdelrahman, Qing Wang, and Bernardo Nunes. “Knowledge tracing: A survey”. In: *ACM Computing Surveys* 55.11 (2023), pp. 1–37.

[2] Fangzhe Ai et al. “Concept-aware deep knowledge tracing and exercise recommendation in an online learning system.” In: *International Educational Data Mining Society* (2019).

[3] John R Anderson, C Franklin Boyle, and Brian J Reiser. “Intelligent tutoring systems”. In: *Science* 228.4698 (1985), pp. 456–462.

[4] Anirudhan Badrinath, Frederic Wang, and Zachary Pardos. “pybkt: An accessible python library of bayesian knowledge tracing models”. In: *arXiv preprint arXiv:2105.00385* (2021).

[5] Ryan SJ d Baker, Albert T Corbett, and Vincent Aleven. “More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing”. In: *Intelligent Tutoring Systems: 9th International Conference, ITS 2008, Montreal, Canada, June 23-27, 2008 Proceedings 9*. Springer. 2008, pp. 406–415.

[6] Joseph E Beck et al. “Does help help? Introducing the Bayesian Evaluation and Assessment methodology”. In: *Intelligent Tutoring Systems: 9th International Conference, ITS 2008, Montreal, Canada, June 23-27, 2008 Proceedings 9*. Springer. 2008, pp. 383–394.

[7] Sven Behnke. “Discovering hierarchical speech features using convolutional non-negative matrix factorization”. In: *Proceedings of the International Joint Conference on Neural Networks, 2003*. Vol. 4. IEEE. 2003, pp. 2758–2763.

[8] Abraham Berman and Robert J Plemmons. “Inverses of nonnegative matrices”. In: *Linear and Multilinear Algebra* 2.2 (1974), pp. 161–172.

[9] Abraham Berman and Robert J Plemmons. *Nonnegative matrices in the mathematical sciences*. SIAM, 1994.

[10] Daniel Billsus, Michael J Pazzani, et al. “Learning collaborative information filters.” In: *Icml*. Vol. 98. 1998, pp. 46–54.

[11] Jason Brownlee. “A gentle introduction to cross-entropy for machine learning”. In: *Machine learning mastery* 20 (2019).

[12] Benoît Choffin et al. “DAS3H: modeling student learning and forgetting for optimally scheduling distributed practice of skills”. In: *arXiv preprint arXiv:1905.06873* (2019).

[13] Irene-Angelica Chounta et al. “Modeling the Zone of Proximal Development with a Computational Approach.” In: *EDM* 2017 (2017), pp. 56–57.

[14] Albert T Corbett and John R Anderson. “Knowledge tracing: Modeling the acquisition of procedural knowledge”. In: *User modeling and user-adapted interaction* 4 (1994), pp. 253–278.

[15] Albert T Corbett and John R Anderson. “Knowledge tracing: Modeling the acquisition of procedural knowledge”. In: *User modeling and user-adapted interaction* 4 (1994), pp. 253–278.

[16] Chris HQ Ding, Tao Li, and Michael I Jordan. "Convex and semi-nonnegative matrix factorizations". In: *IEEE transactions on pattern analysis and machine intelligence* 32.1 (2008), pp. 45–55.

[17] Julian Eggert and Edgar Korner. "Sparse coding and NMF". In: *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*. Vol. 4. IEEE. 2004, pp. 2529–2533.

[18] Susan E Embretson and Steven P Reise. *Item response theory*. Psychology Press, 2013.

[19] Theophile Gervet et al. "When is deep learning the best approach to knowledge tracing?" In: *Journal of Educational Data Mining* 12.3 (2020), pp. 31–54.

[20] Yifan Hu, Yehuda Koren, and Chris Volinsky. "Collaborative filtering for implicit feedback datasets". In: *2008 Eighth IEEE international conference on data mining*. Ieee. 2008, pp. 263–272.

[21] Sheng Jiang, Jianping Li, and Wang Zhou. "An Application of SVD++ Method in Collaborative Filtering". In: *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. IEEE. 2020, pp. 192–197.

[22] Weijie Jiang, Zachary A Pardos, and Qiang Wei. "Goal-based course recommendation". In: *Proceedings of the 9th international conference on learning analytics & knowledge*. 2019, pp. 36–45.

[23] Tanja Käser et al. "Dynamic Bayesian networks for student modeling". In: *IEEE Transactions on Learning Technologies* 10.4 (2017), pp. 450–462.

[24] Mohammad Khajah et al. "Integrating latent-factor and knowledge-tracing models to predict individual differences in learning." In: *EDM*. London. 2014, pp. 99–106.

[25] Yehuda Koren. "Factorization meets the neighborhood: a multifaceted collaborative filtering model". In: *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2008, pp. 426–434.

[26] Yehuda Koren, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems". In: *Computer* 42.8 (2009), pp. 30–37.

[27] Gerd Kortemeyer and Stefan Dröschler. "A user-transaction-based recommendation strategy for an educational digital library". In: *International Journal on Digital Libraries* 22 (2021), pp. 147–157.

[28] Gerd Kortemeyer et al. "Experiences using the open-source learning content management and assessment system LON-CAPA in introductory physics courses". In: *American Journal of Physics* 76.4 (2008), pp. 438–444.

[29] Andrew S Lan and Richard G Baraniuk. "A Contextual Bandits Framework for Personalized Learning Action Selection." In: *EDM*. 2016, pp. 424–429.

[30] Daniel Lee and H Sebastian Seung. "Algorithms for non-negative matrix factorization". In: *Advances in neural information processing systems* 13 (2000).

[31] Daniel Lee and H Sebastian Seung. "Algorithms for non-negative matrix factorization". In: *Advances in neural information processing systems* 13 (2000).

[32] Daniel D Lee and H Sebastian Seung. "Learning the parts of objects by non-negative matrix factorization". In: *Nature* 401.6755 (1999), pp. 788–791.

[33] Greg Linden, Brent Smith, and Jeremy York. "Amazon. com recommendations: Item-to-item collaborative filtering". In: *IEEE Internet computing* 7.1 (2003), pp. 76–80.

[34] Qi Liu et al. "A survey of knowledge tracing". In: *arXiv preprint arXiv:2105.15106* (2021).

[35] Jie Lu et al. "Recommender system application developments: a survey". In: *Decision support systems* 74 (2015), pp. 12–32.

[36] Koki Nagatani et al. "Augmenting knowledge tracing by considering forgetting behavior". In: *The world wide web conference.* 2019, pp. 3101–3107.

[37] Pentti Paatero and Unto Tapper. "Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values". In: *Environmetrics* 5.2 (1994), pp. 111–126.

[38] Shalini Pandey and George Karypis. "A self-attentive model for knowledge tracing". In: *arXiv preprint arXiv:1907.06837* (2019).

[39] Zachary A Pardos and Neil T Heffernan. "KT-IDEM: Introducing item difficulty to the knowledge tracing model". In: *User Modeling, Adaption and Personalization: 19th International Conference, UMAP 2011, Girona, Spain, July 11-15, 2011. Proceedings 19.* Springer. 2011, pp. 243–254.

[40] Zachary A Pardos and Neil T Heffernan. "Modeling individualization in a bayesian networks implementation of knowledge tracing". In: *User Modeling, Adaptation, and Personalization: 18th International Conference, UMAP 2010, Big Island, HI, USA, June 20-24, 2010. Proceedings 18.* Springer. 2010, pp. 255–266.

[41] Philip I Pavlik Jr, Hao Cen, and Kenneth R Koedinger. "Performance Factors Analysis–A New Alternative to Knowledge Tracing." In: *Online Submission* (2009).

[42] Chris Piech et al. "Deep knowledge tracing". In: *Advances in neural information processing systems* 28 (2015).

[43] P Resnick. "Anopen architecture for collaborative filterring of netnews". In: *Proc CSCW'94.* 1994.

[44] Paul Resnick and Hal R Varian. "Recommender systems". In: *Communications of the ACM* 40.3 (1997), pp. 56–58.

[45] Cristóbal Romero and Sebastián Ventura. "Educational data mining: a review of the state of the art". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (applications and reviews)* 40.6 (2010), pp. 601–618.

[46] Sheldon M Ross. *Introduction to probability models.* Academic press, 2014.

[47] Badrul Sarwar et al. "Item-based collaborative filtering recommendation algorithms". In: *Proceedings of the 10th international conference on World Wide Web.* 2001, pp. 285–295.

[48] Rashish Tandon and Suvrit Sra. "Sparse nonnegative matrix approximation: new formulations and algorithms". In: (2010).

[49] Wei Xu, Xin Liu, and Yihong Gong. "Document clustering based on non-negative matrix factorization". In: *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval.* 2003, pp. 267–273.

[50] Chun-Kit Yeung. "Deep-IRT: Make deep learning based knowledge tracing explainable using item response theory". In: *arXiv preprint arXiv:1904.11738* (2019).

[51] Chun-Kit Yeung and Dit-Yan Yeung. "Addressing two problems in deep knowledge tracing via prediction-consistent regularization". In: *Proceedings of the fifth annual ACM conference on learning at scale.* 2018, pp. 1–10.

[52] Michael V Yudelson, Kenneth R Koedinger, and Geoffrey J Gordon. "Individualized bayesian knowledge tracing models". In: *Artificial Intelligence in Education: 16th International Conference, AIED 2013, Memphis, TN, USA, July 9-13, 2013. Proceedings 16.* Springer. 2013, pp. 171–180.

[53] Jiani Zhang et al. “Dynamic key-value memory networks for knowledge tracing”. In: *Proceedings of the 26th international conference on World Wide Web*. 2017, pp. 765–774.

[54] Yunhong Zhou et al. “Large-scale parallel collaborative filtering for the netflix prize”. In: *Algorithmic Aspects in Information and Management: 4th International Conference, AAIM 2008, Shanghai, China, June 23-25, 2008. Proceedings 4*. Springer. 2008, pp. 337–348.