# Tensor Factorization for Student Modeling and Performance Prediction in Unstructured Domain

**Shaghayegh Sahebi** SHS106@PITT.EDU
Intelligent systems Program, University of Pittsburgh, Pittsburgh, PA

**Yu-Ru Lin** YURULIN@PITT.EDU
School of Information Sciences, University of Pittsburgh, Pittsburgh, PA

**Peter Brusilovsky** PETERB@PITT.EDU
School of Information Sciences, University of Pittsburgh, Pittsburgh, PA

## Abstract

We propose a novel tensor factorization approach, Feedback-Driven Tensor Factorization (FDTF), for modeling students' learning process and predicting student performance. This model decomposes the tensor built upon students' attempt sequence, considering the quizzes students select to work with as its feedback. FDTF does not require any prior domain knowledge, such as learning resource skills, concept maps, or Q-matrices. Also, our proposed model differs from other tensor factorization approaches as it explicitly models the constant learning of students while interacting with the learning resources. We compare our approach with other state-of-the-art approaches in the task of Predicting Student Performance (PSP). Our experiments show that FDTF performs significantly better compared to Bayesian Knowledge Tracing and baseline tensor factorization algorithm.

## 1. Introduction

The growth of Massive Open Online Courses (MOOC) rapidly increased the volume of data on students' education and learning behavior. This abundance of data calls for approaches that can automatically make sense of such data, avoiding the need for manual handling of it. Predicting students performance and modeling student knowledge are two of the tasks that help to understand such data. The goal in predicting student performance (PSP), is to estimate if a specific target student can handle a learning material successfully, e.g. if the student can succeed or fail in solving a specific question. Student knowledge modeling aims to quantify and approximate student's knowledge at each moment in time in each of the possible existing skills (or concepts) in the learning material. The set of skills are defined, manually or automatically, based on the learning material.

Understanding students attempt data through PSP and student knowledge modeling encourages better course design by teachers, targeted personalization of course pace, and more accurate automatic learning resource recommendation to students. Hence, one of the main foci in the educational data mining literature has been on predicting student performance and student knowledge modeling.

For example, Bayesian Knowledge Tracing was one of the pioneer approaches that could predict success or failure of students in problems (Corbett & Anderson, 1994). Recently, other approaches, such as factorization models have been used for PSP. For example, Performance Factor Analysis (PFA) (Pavlik Jr et al., 2009) is another approach to PSP and cognitive modeling. PFA takes into account the effects of the initial difficulty of the skills (knowledge components) and prior successes and failures of a student on those skills associated with the current item. These approaches require to know the structure of learning domain, as a domain model, or the association between skills and learning material, a priori.

More recent approaches have aimed to overcome this limitation by using approaches such as latent factor models. For example, Thai-Nghe et al. experimented

*Proceedings of the 33$^{rd}$ International Conference on Machine Learning*, New York, NY, USA, 2016. JMLR: W&CP volume 48. 

on a context-aware factorization algorithm, based on collaborative filtering approaches in the recommender system literature (Thai-Nghe et al., 2011b). Sahebi et al. studied various methods of educational data mining field along with matrix and tensor factorization approaches, from the recommender systems literature, for PSP (Sahebi et al., 2014a). Lan et al. used quantized matrix completion to predict students' performance in SPARFA-Lite (Lan et al., 2014).

Tensors, or multi-dimensional matrices, have been used in the literature to represent student attempt data (Sahebi et al., 2014b; Thai-Nghe et al., 2011a). One of the main reasons for using tensors to represent educational data is their seamless flexibility in representing multiple dimensions of data, such as students, questions, time, and topic structure. Another reason for using tensors is their decomposition adaptability to the task at hand.

While there are various tensor decomposition models and algorithms in the literature (Kolda & Bader, 2009), the potential to versatile modeling of tensors in the educational data mining field is under-explored. Although using generic tensor factorization models, compared to the traditional PSP approaches, results in better, or comparable, performance (Sahebi et al., 2014b; Thai-Nghe et al., 2011a), these tensor models are not tailored for the educational data. More specifically, these models are built for other purposes than educational data mining (such as recommender systems), and thus do not consider the characteristics of educational data mining challenges.

One of these challenges is the knowledge increase of students while interacting with learning material. As the students interact with quizzes, readings, and other learning resources, they incrementally learn the underlying skills that are present in these resources. Also, this amount of knowledge increase for a student depends on the material that this students is interacting with. The current tensor factorization approaches that are used for PSP in the literature do not model this interaction.

In this paper, we provide a solution to this problem by proposing a special form of tensor factorization model that can take into account the constant learning of students. Our proposed tensor factorization model, called feedback-driven tensor factorization, directly models the knowledge increase of students by adding a feedback based constraint on the previous student's knowledge and the current learning material student is using. This approach does not require any domain knowledge, and can be used in unstructured domains. We compare our approach with Bayesian Knowledge Tracing and a baseline tensor factorization algorithm. Our experiments show the superior performance of our proposed approach, compared to the baseline methods.

## 2. Feedback-Driven Tensor Factorization (FDTF)

As mentioned in the introduction, the goal of our approach is to predict the students performance considering the constant learning of students. In order to achieve this goal, we represent student activities on learning material as a three-dimensional tensor $\mathcal{Y}$[1].

Suppose that the students are working on one resource type and learning from it. To be more specific, suppose that $m$ students are interacting with $n$ quizzes and each student can have multiple attempts (at most $l$) on each quiz. Then, we can model the students' attempt sequences on all quizzes as a tensor of size $m \times n \times l$. The $k^{th}$ frontal slice of this tensor ($\mathcal{Y}_{:,:,k}$) shows the success or failure of all students on all quizzes in their $k^{th}$ attempt. For abbreviation, we use $\mathcal{Y}_k$ for representing the $k^{th}$ frontal slice of all tensors. Accordingly, $\mathcal{Y}_{i,:,:}$ shows all the attempts of student $i$ on all questions and $\mathcal{Y}_{:,j,:}$ shows all attempts of all students on question $j$. We assume that each quiz is consisted of multiple ($c$) concepts (skills or knowledge components) and the students should have some knowledge on these concepts to be able to solve the quizzes that include these concepts in them. Some of the elements of $\mathcal{Y}$ are unknown to us because not all of the students try all of the questions as many times. Based on these assumptions, we model the problem as a tensor factorization algorithm with two phases: the prediction phase and the learning phase.

In the prediction phase, we follow the assumption that students' success or failure in quizzes depends on their knowledge and the concepts underlying those quizzes. In this phase, we decompose $\mathcal{Y}$ into a tensor and a matrix: the tensor $\mathcal{T}$ that shows the knowledge of students on the concepts at each of their attempts on the quizzes, and the matrix $\mathbf{Q}$ that shows the concepts required to solve each quiz correctly. For each quiz $j$, $\mathbf{Q}_{:,j}$ shows the importance of each of the discovered concepts in it. Also, $\mathcal{T}_{i,k,l}$ shows the knowledge of student $i$ in concept $k$ at the $l^{th}$ attempt.

Based on this decomposition, we can estimate (pre-

[1] In this paper, tensors are represented by script letters, e.g. $\mathcal{Y}$; Matrices are denoted by boldface capital letters, e.g. $\mathbf{X}$; and vectors are represented by boldface lowercase letters, e.g. $\mathbf{x}$. In addition, we denote the $i^{th}$ row of a matrix $\mathbf{X}$ as $\mathbf{X}_{i,:}$, the $j^{th}$ column as $\mathbf{X}_{:,j}$, and the entry $(i,j)$ as $\mathbf{X}_{i,j}$

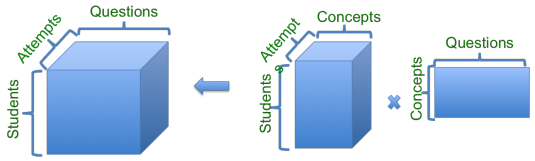


*Figure 1.* Phase 1: Decomposition of Students' Performance into Students' Knowledge and Concept-Map

dict) the unknown values of $\mathcal{Y}$ using multiplication of tensor $\mathcal{T}$ and matrix $\mathbf{Q}$ as presented in Equation 1. Figure 1 is an illustration of this decomposition.

$$\mathcal{Y} = \mathcal{T} \times \mathbf{Q} \tag{1}$$

We suppose that students learn by practicing the quizzes and the knowledge of students increase by this practice over the concepts. The learning phase of our tensor factorization approach models learning of students based on the quizzes that they choose to solve in each step. In order to do that, we build a tensor $\mathcal{X}$ that denotes if a student has chosen to work on a specific problem, at a specific time, or not. Equation 2 shows how to build this tensor based on $\mathcal{Y}$.

$$\mathcal{X}_{i,j,k} = \begin{cases} 1, & \text{if } \mathcal{Y}_{i,j,k} \text{ is observed} \\ 0, & \text{if } \mathcal{Y}_{i,j,k} \text{ is not observed} \end{cases} \tag{2}$$

In the learning phase, we assume that the amount of gained knowledge in each concept is a function of student's knowledge at the previous attempt and the weight of concepts that are learned in the quiz that is selected to be solved by the student. Let $f(\cdot)$ be such function; then the gained knowledge at time $t$ can be expressed as:

$$\mathcal{T}_t = f(\mathcal{T}_{t-1}, \mathcal{X}_t, \mathbf{Q})$$

Since we assume that knowledge of students grows over time, we should choose a monotonically increasing function for $f(\cdot)$. Also, to avoid this knowledge increase to grow too large, this function should be bounded. Based on these assumptions, we model the knowledge growth of students as a logistic regression function ranging between 0 (for no increase in the knowledge) to $1 - \mathcal{T}_{t-1}$ (for the maximum increase in the knowledge). This allows us to have a bounded amount of knowledge that always stays between zero and one. To add to the flexibility of this function, and account for different students' rate for learning from the quizzes, we add a factor $\mu$ that controls the slope of the logistic regression function. The more learning rate ($\mu$) is, the bigger the knowledge increase will be and the students get to maximum state of knowledge faster. This increase can be seen in Equation 3.

$$\mathcal{T}_t = \mathcal{T}_{t-1} + \left(\frac{2(1 - \mathcal{T}_{t-1})}{1 + exp(-\mu \mathcal{X}_t \mathbf{Q}')} - (1 - \mathcal{T}_{t-1})\right), \tag{3}$$

which can be written as follows:

$$\mathcal{T}_t = 2\mathcal{T}_{t-1} + \frac{2(1 - \mathcal{T}_{t-1})}{1 + exp(-\mu \mathcal{X}_t \mathbf{Q}')} - 1 \tag{4}$$

Based on this model, the more knowledgeable the student is in a concept, the less improvement she will get by practicing the same concepts again and again. The most increase in the student's knowledge happens when the student does not know the skills provided in the quiz. If we expand and simplify the Equation 3, we achieve Equation 4. Since $f(\cdot)$ is a monotonically increasing function, the estimated knowledge tensor ($\mathcal{T}$) and latent factors ($\mathbf{Q}$) are both non-negative. This non-negativity is in accordance with assumptions in educational domain: that the knowledge of students at any time over any concept cannot be negative.

Eventually, the matrix factorization includes solving Equations 1 and 4. Assuming we have the values for $\mathcal{X}_t$ and $\mathbf{Q}$, Equation 4 can be considered as a static update and we can only optimize Equation 1 iteratively and update the knowledge values in each iteration using Equation 4. To this end, we try to optimize for the least regularized estimation error of our observed tensor ($\mathcal{Y}$) in Equation 5. Thus, our objective is to minimize the overall error, defined as:

$$\Sigma_{i=1}^{t} \parallel \mathcal{Y}_t - \mathcal{T}_t \mathbf{Q} \parallel^2 + \lambda(\Sigma_{i=1}^{t} \parallel \mathcal{T}_i \parallel^2 + \parallel \mathbf{Q} \parallel^2) \tag{5}$$

The last two terms are added to the error equation to regularize the values in tensor $\mathcal{T}$ and matrix $\mathbf{Q}$. These two terms increase the sparsity of the knowledge model and latent factor matrix, by decreasing the values in these two factors, while preventing the factorization to overfit to the training data.

Since this method is using the iterative feedback loops and the two phases of prediction and learning, we name it Feedback-Driven Tensor Factorization (FDTF).

## 3. Experiments

To asses the student performance prediction task, we compare the proposed FDTF model with a baseline tensor factorization algorithm that is introduced in recommender system literature. This tensor factorization algorithm is called Bayesian Probabilistic Tensor Factorization (BPTF) and models the temporal

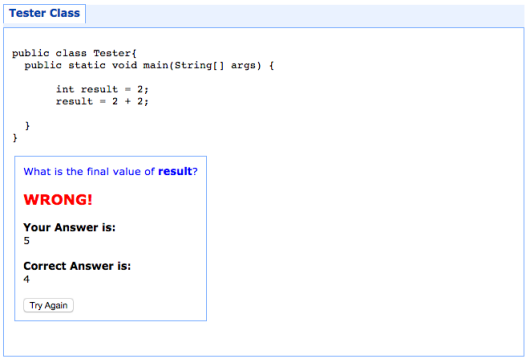


*Figure 2.* Screen-shot of QuizJet System

change of user interests on items (Xiong et al., 2010). We choose this model as a baseline because of its consideration of time sequencing and the common use of recommender systems algorithms in the educational data mining literature (Sahebi et al., 2014a). As our second baseline, we run Bayesian Knowledge Tracing (BKT) algorithm on the data (Corbett & Anderson, 1994). Since BKT requires a pre-defined set of concepts, we use the manually-labeled concepts discovered by experts in this case.

The FDTF algorithm has two parameters that need to be tuned: the number of concepts ($c$) and the learning rate of students ($\mu$). We define these two parameters by cross-validation.

### 3.1. Dataset and Setup

We use student sequences of QuizJet online self-assessment system to run our experiments (Hsiao et al.). This system produces parameterized Java quizzes based on a set of predefined templates. Hence, each student can repeat the same Java quiz, with different parameters, over and over again. The students submit their answer using a text box provided in the user interface and receive an immediate feedback. Figure 2 shows a screen-shot of this system.

The dataset is collected from the students who have taken Java programming course from Fall 2010 to Spring 2013 (six semesters). The system was introduced in the class and students have interacted with this system voluntarily. The subject domain is organized by experts into 22 coherent topics. Each topic has several questions and each question is assigned to one topic. We use these sets of topics as the expert-labeled domain model in our experiments.

We experimented on $27,302$ records of 166 students on 103 questions. The average number of attempts on each question is equal to three. Our dataset is imbalanced: the total number of successful attempts in the data equals to $18,848$ (69.04%) and the total number of failed attempts is 8454. We used user-stratified 5-fold cross-validation to split the data, so that the training set has 80% of the users (with all their records) randomly selected from original dataset, while the remaining 20% of the users were retained for testing. In other words, 80% of students are in training set: we have all of their sequences. For the remainder of students (the 20%) we use 20% of their data to predict the rest 80% of it. Eventually, we have $80\% + 20\% * 20\% = 84\%$ of the whole dataset in the training set. We use the same set of data for all of the algorithms. We run the experiments 3 times per stratification and end up with running each algorithm 15 times. Simple statistics of our dataset are shown in Table. 1.

To find the best number of concepts ($c$) in each of the automatic PSP algorithms, we use cross-validation.

### 3.2. Experimental Results

As explained in Section 3, we examine the prediction performance of the proposed FDTF algorithm and the baseline models: BPTF and BKT with expert-labeled topics. Figure 4 shows the accuracy of the mentioned algorithms. The red, green, and cyan bars represent accuracy of FTDF, BPTF, and BKT. As we can see in this figure, Although accuracy of the baseline tensor factorization model (BPTF) is better than Bayesian Knowledge Tracing, it is significantly less than the accuracy of the proposed approach (FDTF). Eventually, FDTF performs significantly better than both of the baseline algorithms.

Although the task of predicting students performance is a binary classification task in this setting (predicting failure or success for students), the Root Mean Squared Error (RMSE) is traditionally used for evaluation of this task in the literature. As a result, we compare the approaches based on RMSE of approaches in addition to accuracy. Figure 3 shows RMSE of these experiments for each of the approaches. Again, we can see that FDTF has a significantly better RMSE than both BKT and BPTF algorithms.

These results show that, even though BKT has the additional knowledge of topic-based domain model, the tensor factorization algorithms outperform it. Additionally, despite the fact that both BPTF and FDTF use the same data, model the student data as a tensor, and are temporal tensor factorization approaches, the proposed FDTF approach performs better than BPTF. These results show that explicitly modeling students' knowledge acquisition, by considering their interactions with learning materials, leads to better

*Table 1.* Dataset Statistics

| | Average | Min | Max |
|---|---|---|---|
| #attempts per sequence | 3 | 1 | 50 |
| #attempts per question | 265 | 25 | 582 |
| #attempts per student | 165 | 2 | 772 |
| #different students per question | 87 | 7 | 142 |
| #different questions per student | 54 | 1 | 101 |

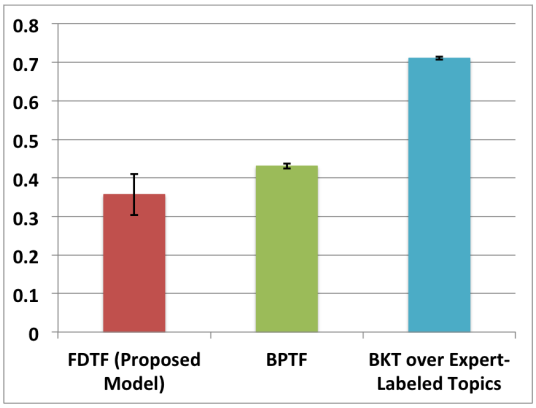


*Figure 3.* RMSE of Algorithms for Predicting Students Performance

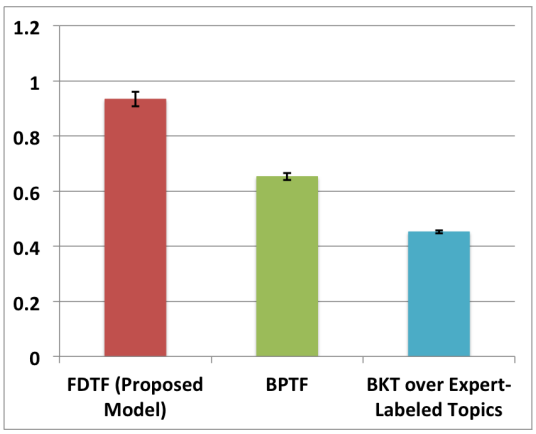


*Figure 4.* Accuracy of Algorithms for Predicting Students Performance

modeling of student knowledge, and thus better prediction of student performance.

## 4. Conclusion and Future Work

We proposed a novel tensor factorization model (FDTF) that can predict students' success or failure in the future quizzes, by explicitly modeling their knowledge acquisition during interaction with learning material. This approach does not require any expert or domain knowledge and can automatically perform using students' historical attempt sequence. Our evaluations show that FDTF outperforms the predicting student performance approaches in the literature.

In future, We plan to explore the ability of the proposed approach in discovering the underlying domain model for the learning material, experiment on more diverse datasets, and compare our algorithm to other PSP and domain modeling approaches in the literature. We would like to improve FDTF to be able to model implicit feedback of students' activity, in addition to success and failure records of them.

The FDTF model has the potential to be used as a basis for recommendation of learning material to students. Also, it can be help teachers to discover domain model and edit the learning material, look up the concepts students are weak at, and suggest the appropriate learning activities to students.

## References

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

Hsiao, I-Han, Sosnovsky, Sergey, and Brusilovsky, Peter. Adaptive navigation support for parameterized questions in object-oriented programming. In *ECTEL 2009*, volume 5794 of *LNCS*, pp. 88–98. Springer-Verlag.

Kolda, Tamara G and Bader, Brett W. Tensor de-

compositions and applications. *SIAM review*, 51(3): 455–500, 2009.

Lan, Andrew S, Studer, Christoph, and Baraniuk, Richard G. Quantized matrix completion for personalized learning. 2014.

Pavlik Jr, Philip I, Cen, Hao, and Koedinger, Kenneth R. Performance factors analysis–a new alternative to knowledge tracing. *Online Submission*, 2009.

Sahebi, Shaghayegh, Huang, Yun, and Brusilovsky, Peter. Predicting student performance in solving parameterized exercises. In *Intelligent Tutoring Systems*, pp. 496–503. Springer, 2014a.

Sahebi, Shaghayegh, Huang, Yun, and Brusilovsky, Peter. Parameterized exercises in java programming: using knowledge structure for performance prediction. In *The second Workshop on AI-supported Education for Computer Science (AIEDCS)*, pp. 61–70. University of Pittsburgh, 2014b.

Thai-Nghe, Nguyen, Drumond, Lucas, Horváth, Tomás, Nanopoulos, Alexandros, and Schmidt-Thieme, Lars. Matrix and tensor factorization for predicting student performance. In *CSEDU (1)*, pp. 69–78. Citeseer, 2011a.

Thai-Nghe, Nguyen, Horvath, Tomas, and Schmidt-Thieme, Lars. Context-aware factorization for personalized student's task recommendation. In *Proceedings of the International Workshop on Personalization Approaches in Learning Environments*, volume 732, pp. 13–18, 2011b.

Xiong, Liang, Chen, Xi, Huang, Tzu-Kuo, Schneider, Jeff G, and Carbonell, Jaime G. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*, volume 10, pp. 211–222, 2010.