Modeling Multiple Subskills by Extending Knowledge Tracing Model Using Logistic Regression

Xuan Zhou  
State Key Lab of Software Development Environment  
Department of Computer Science and Engineering,  
Beihang University  
Beijing, China  
zhouxuan@nlsde.buaa.edu.cn

Wenjun Wu, Yong Han  
State Key Lab of Software Development Environment  
Department of Computer Science and Engineering,  
Beihang University  
Beijing, China  
{wwj, hanyong}@nlsde.buaa.edu.cn

Abstract—Knowledge Tracing (KT) is a standard model for inferring student knowledge mastery from their performance data. Generally, there are five parameters in KT model we need to estimate, they are transition parameters: learning and forgetting probability, emission parameters: guessing and slipping probability, and prior probability. KT model is widely used in students learning outcome modeling. However, it does not support multiple subskills modeling, for the model works by checking the historical observations at a specific skill. To overcome this drawback, this paper proposes three models using logistic regression over each step: KTLR-GS, extending guessing and slipping parameters; KTLR-LFID, introducing item difficult into the KT model while extending learning and forgetting parameters; KTLR-FP, extending both transition and emission parameters. Unlike previous methods our models discuss the efficiency of extending transition and emission parameters while relaxing the assumptions that subskills are independent. In the end we evaluate how well our models perform with comparison to LR-DBN, KT-IDEM and KT on two datasets: the open dataset ASSISTments, and one private dataset, student algebra dataset collected through our tutoring system. Our models outperform LR-DBN, KT-IDEM and KT on both datasets.

Keywords: knowledge tracing; logistic regression; multiple subskills.

I. INTRODUCTION

Various kinds of e-learning systems, such as Massively Open Online Courses[1] and intelligent tutoring systems, are now producing large amounts of feature-rich data from students solving items at different levels of proficiency over time. To analyze such educational big data, researchers often use Knowledge Tracing Model [2] (KT), a 20-year-old method that had become the de-facto standard for inferring student knowledge mastery through their performance on exercise items. There are five parameters in KT model, the transmission parameters learning and forgetting probability, the emission parameters guessing and slipping probability, and the prior probability that a student already know the skill. Unfortunately, the original KT model assumes each exercise item only involves single skill without considering latent multiple subskills in student homework and assignments.

Previous research has explored various approaches to this problem. Generally speaking, there are three groups including conjunctive models, performance factor analysis and logistic regression based models. Conjunctive models assume that a student must master all of the subskills in order to perform the step correctly [3][4]. They calculate the probability of knowing them all by multiplying the probabilities of knowing the individual subskills. Learning factor analysis models capture the knowledge components of courses as features, take the form of logistic regression modelling student performance [10]. Logistic regression based models uses logistic regression to extend parameters in KT model separately [13][14].

However, there are several weaknesses of the models above. Conjunctive models and learning factor analysis models don’t consider the time sequence of the questions answered. Logistic regression models haven’t fully investigated the effect of transmission and emission parameters in KT model.

This paper presents three new ways to extend KT model using logistic regression to trace multiple subskills in a Bayesian network without assuming they are independent. They are named KTLR-GS, KTLR-LFID and KTLR-FP. In KTLR-GS, we extend the emission parameters in logistic regression. In KTLR-LFID we extend the transmission parameters in logistic regression, in addition, we individualize the emission parameter with item difficulty. KTLR-FP combines the first two models extending transmission and emission parameters. We have conducted extensive experiments to evaluate the performance of these models and compare them with previous research models, especially those that also adopt logistic regression based subskill modeling. Experimental analytic results confirm the performance improvement of our models. Furthermore, the detailed analysis of the parameter estimation process based on Expectation Maximization algorithm for our models indicates that the right strategy to choose the models for different datasets.

The rest of this paper is organized as follows: Section 2 introduces the KT model, and current research on multiple subskills modeling. Section 3 describes our three models. Section 4 introduces the datasets we used in this paper. Section 5 reports the experimental results of our three models [14]; Section 6 concludes our current work and discusses future work.
II. RELATED WORK

A. Knowledge Tracing

In the Intelligent Tutoring System (ITS) [6], there are a variety of methods to model students learning outcomes. The Knowledge Tracing Model [2] shown in Figure 1 has been widely used. The standard KT model has five parameters that are learned from data for each given skill. These five parameters indicate the model’s deduced probability whether the student has mastered the skill given his chronological sequence of correct and incorrect responses to a particular set of questions. P(G) and P(S) are two parameters that determine the student’s performance on a question based on his current mastery of this knowledge. P(T) is the guessing parameter that a student will answer correctly even if he doesn’t know this skill to the question. P(S) is the slipping parameter that a student will answer incorrectly when he already knows this skill. P(T) and P(F) parameters dictate the change between the states of the student knowledge nodes. P(T) is the learning parameter indicating the probability of transmitting from not mastering to mastering the skill. P(F) is the forgetting rate describing the probability of forgetting this skill transmitting from knowing to not knowing. P(L0) is referred as the prior probability of the initial knowledge node where a student knows the skill before answering the first question.

Since the KT model has been proposed, it has been broadly used to evaluate and predict student knowledge mastery. There are some other extensions of the KT model. For example, KT-IDEM [23] adds the item difficulty while keeping the modification limited to slight changes to the topology of the KT model. However, when it comes to multiple subskills, KT isn’t able to handle all the skills simultaneously, because the KT model works by checking the historical observations on a specific skill. Many research efforts have been done to address this problem by incorporating multiple skills factors in their models. They can be roughly categorized into three major groups: conjunctive models, performance factor analysis and logistic regression based models.

![Figure 1 The Standard Knowledge Tracing Model](image)

B. Conjunctive Subskill Modeling

The conjunctive models, assume students must master all the subskills to have a grasp of one specific skill. In [2], Gong proposes two approaches to this problem. The first approach is inspired by the joint probability in Probability Theory. It assumes that the latent subskills for each exercise item are independent and describes the overall probability of knowing the right answer to the item as the multiplication of all the probabilities of every individual subskill. However, such as a probability product usually underestimates the probability of one student knowing all the subskills. The minimum of the estimated probability provides a less pessimistic estimate based on the assumption that the likelihood of a correct answer is dominated by the student’s weakest subskill.

The other approach proposed by Gong uses the minimum of the probability of all the subskills to represent the probability of mastering this skill. However, when the difference between the minimum and maximum is very large, using the minimum to represent the student’s probability does not seem convincing.

Inspired by the Item Response Theory (IRT) [17], Cen [4] models the conjunction as a multiplication of subskills, in which each subskill has a parameter representing its difficulty discrimination and other parameters representing proficiency of an student. It’s a special case of Embretson’s multicomponent latent trait model [6] and customized for the high dimensional feature of intelligent tutoring system. However, it does not trace skills over a time slice.

C. Performance Factor Analysis

Performance Factor Analysis (PFA) is an alternative way for student skill modeling [7]. As a variant of learning decomposition [8], PFA is built on reconfiguring Learning Factor Analysis (LFA) [9].

LFA is a general method for cognitive model that captures the knowledge components in a curriculum [10]. LFA has three important components including a statistical model that determines the skills, the item difficulty that may affects student performance, and a combinatorial search that performs model selection within the logistic regression model space [11]. The statistical model takes the form of logistic regression with student performance as the dependent variable. Item difficulty factors can identify a property of problems that causes student difficulties.

Inspired by LFA, [4] proposes two PFA models, which adopt an additive model instead of conjunctive model. It assumes that the probability of a student answering a question correctly is proportional to the amount of all the required knowledge the students knows plus the “easiness” of the question and the amount of learning gained for each practice opportunity. While the other one combined Item Response Theory and conjunctive model, supposing the multiple subskills modeling is a conjunctive model, which could perform by multiplying all the subskill parameters.

Pavlik [7] reconfigures LFA on model’s independent variables by dropping the student variables and replacing the skill variable with the questions identity. It takes the form of standard logistic regression with student performance as a dependent variable, and estimates a parameter for each question representing the question difficulty, and two parameters for each skill indicating the effects of the prior success and prior failures for that skill [12].

However, when it comes to multiple subskills modeling, models we described above don’t consider the order of the questions occurred.
D. Logistic Regression based Subskill Modeling

All the previous models ignore the fact that when a student answering questions, he must work it out in a specific order. So the temporal learning procedure is an important factor for modeling student performance.

Inspired by the logistic regression over the learning and forgetting probabilities, Gonzalez-Brenes [13] thinks different features, such as subskills, problem's difficulty, and student ability etc, influence parameters in KT model. Each parameter could be expressed in logistic regression with the features we discussed. With different features, the KT model could be presented in different ways. So he proposes a general method that allows efficient general features into KT model named Feature-Aware Student Knowledge Tracing (FAST). Unfortunately, in this paper, Brenes only focuses on the guessing and slipping parameters without describing the extension in the learning and forgetting parameters.

Yanbo [14] restructures the KT model using logistic regression over each step’s subskills to model the learning and forgetting probabilities for overall knowledge required by the step, the model is called LR-DBN. In this paper Yanbo proves the equivalence of this extension and modeling the knowledge probabilities with logistic regression. LR-DBN is applied on the Project LISTEN’ Reading Tutor [15] during 2005-2006 school year. The result shows a particular efficiency in the case of tasks involving multiple subskills [16]. However, this model only extends the transition parameter, lacking the discussion of emission parameters.

Neither of the above logistic regression-based models have fully investigate whether all the parameters including learning rate, forgetting probability, guessing and slipping probability can be integrated in a single model and how to tune these parameters in different subskill distributions. In the next section, we introduce a group of multi-subskill models that integrate all the four parameters and item difficulty through logistic regression in the KT models.

III. EXTENDING KNOWLEDGE TRACING TO MODEL MULTIPLE SUBSKILLS WITH LOGISTIC REGRESSION

In the traditional KT Model, we estimate the parameters depending on the skill that it requires to perform. We use latent variable to model the hidden knowledge state that changes over steps, and infer the state from sequential observations of student’s performance. If we know or assume which set of subskills a step requires, it makes sense to estimate overall knowledge and inference of the step as a function of the estimated knowledge of each individual subskill it requires.

In this section we propose three new models (including KTLR-GS, KTLR-LFID and KTLR-FP) using logistic regression functions.

A. KTLR-GS: Extending Guessing and Slipping Probability In Knowledge Tracing Model Using Logistic Regression

We assume that the guessing and slipping probability on each step are based on the student ability of each subskill. Thus the guessing and slipping probability distinct over each step. The more subskills a step needs, the less possible student could guess the right answer. The architecture of model KTLR-GS is shown in Figure 2. The definition of the variables is listed as follows:

- \( S_{ij}^{(n)} \): observed variable, the subskill used in step n; 1 if step n requires subskill j, 0 otherwise.
- \( K_n \): hidden variable, student knowledge state, define whether a student has mastered knowledge at this step; true if student has known, false otherwise.
- \( C_n \): observed variable, student performance node, defines the result of this step; true if student answers the question correctly, false otherwise.
- \( P(L_n) \): the prior probability, defines the probability that a student knows this skill before he starts answering this set of question.
- \( P(T) \): the learning probability; the probability of a student transferring from not knowing the knowledge in step n to knowing in step n + 1.
- \( P(F) \): the forgetting probability; the probability of a student transferring from knowing knowledge in step n to not knowing in step n + 1. The traditional KT model assumes there are no forgetting, however it is impossible to keep memory of a specific skill without forgetting. So this model uses the forgetting probability to describe forgetting effect.

The guess probability \( P(G) \) in step n is defined in the form logistic regression:

\[
P(C_n = true | K_n = false) = \frac{1}{1+\exp[-\sum_{i=1}^m \alpha_i S_{ni}^{(i)} + \beta]} \tag{1}
\]

Similarly, the slipping probability \( P(S) \) in step n is defined:

\[
P(C_n = false | K_n = true) = \frac{\exp[-\sum_{i=1}^m \delta_i S_{ni}^{(i)} + \gamma]}{1+\exp[-\sum_{i=1}^m \delta_i S_{ni}^{(i)} + \gamma]} \tag{2}
\]

Here \( \alpha_i \) and \( \delta_i \) represent skill j’s respective contributions to the guessing and slipping probability, \( \beta \) and \( \gamma \) is the offset of the guess and slip probability. We assume that \( \alpha_i, \delta_i, \beta, \gamma \) is a reflection of the student’s overall ability in the practice, so they do not change over time.
Unlike KT model, changes of guessing and slipping probability on a specific step \( n \) are not a fixed conditional probability table, they depend on the distribution of the subskills and the coefficient of each subskill. Figure 3 illustrates the changes of guessing probability on the condition when there are only two subskills involved on a specific step. This implies that the group of subskills has a direct impact on both guessing and slipping probability: higher subskill levels results in lower guess probability and higher slip probability.

1) Parameter Estimation: We run Expectation Maximization algorithm (EM algorithm) to infer the parameters, which is a popular approach to estimate the parameters in the KT model. The “E step” uses the current parameters (\( \lambda \)) as the estimation for the transition and emission probabilities to infer the probability that the student has mastered the skill at each practice opportunity. Given the estimated probabilities of mastery computed in the E step, the “M step”, precomputes the parameter (\( \lambda \)) estimates

   a) E step: In this model, the student knowledge nodes are hidden, while the subskills node and student performance node are observed. The student knowledge has two states: knowing and not knowing. The student performance node has two states: answering correctly and not correctly. The subskills node also has two values, 0 means this step doesn’t contain this subskill, while 1 means this subskill was inspected in this step.

   We use \( i_{T} \) to represent the state of knowledge node in step \( T \), \( a_{i_{T|i_{T+1}}} \) represent the probability of transferring from state \( i_{T} \) to state \( i_{T+1} \). When \( i_{T} = 0 \) and \( i_{T+1} = 1, \) \( a_{i_{T|i_{T+1}}} \) represents the learning probabilities, which means \( P(T) \). When \( i_{T} = 1 \) and \( i_{T+1} = 0, \) \( a_{i_{T|i_{T+1}}} \) represents the forgetting probabilities, which means \( P(F) \). \( \pi_{0i} \) means the prior probability \( P(L_{0}) \).

   We use \( b_{i_{T}}(O_{T}) \) to represent the probability of when knowledge state is \( i_{T} \) and the performance state is \( O_{T} \) in the step \( T \). When \( i_{T} = 0 \) and \( O_{T} = 1, \) \( b_{i_{T}}(O_{T}) \) means the guessing probability \( P(G) \) in step \( n \). When \( i_{T} = 1 \) and \( O_{T} = 0, \) \( b_{i_{T}}(O_{T}) \) means the slipping probability \( P(S) \) in step \( n \). In this condition, \( P(G) \) could be inferred as Eq(1), \( P(S) \) could be inferred as Eq(2).

   The expected value of the log likelihood function is

   \[
   Q(\lambda, \tilde{\lambda}) \equiv \sum_{i} \log P(O, I | \lambda)P(O, I | \tilde{\lambda})
   \]  

   “O” represents the observation sequence, “I” represents the hidden states. “\( \lambda \)” represents the parameters described above. “\( \tilde{\lambda} \)” represents the current estimated parameters.

   \[
P(O, I | \lambda) = \pi_{i_{0}} b_{i_{0}}(O_{0})a_{i_{1}} b_{i_{1}}(O_{1})a_{i_{2}} \ldots a_{i_{n}} b_{i_{n}}(O_{n})
   \]  

So Eq(3) could be written as:

   \[
   Q(\lambda, \tilde{\lambda}) = \sum_{i} \log \pi_{i} P(O, I | \lambda) + \sum_{i} \left( \sum_{n=1}^{N} \log a_{i_{n}} a_{i_{n+1}} \right) P(O, I | \tilde{\lambda}) + \sum_{i} \left( \sum_{n=1}^{N} \log b_{i_{n}}(O_{n}) \right) P(O, I | \tilde{\lambda})
   \]

   b) M step: In M step we need to find the parameter to maximize the log likelihood function. So we need to find the correct \( \lambda \) to maximize the function: \( \lambda = \arg \max_{\lambda} Q(\lambda, \tilde{\lambda}) \). In order to find the correct parameters to maximize the log likelihood function, we have to take the derivative of Eq(5).

   As we have split Eq(3) into three parts, each part has specific parameters. So we only need to take the derivative of the part where the parameter we want to estimate is in. For example, to calculate derivation of \( \pi_{10} \):

   \[
   \frac{\partial Q}{\partial \pi_{i}} = \frac{\partial}{\partial \pi_{i}} \left( \sum_{n=1}^{N} \log \pi_{i} P(O, i_{n} = | \lambda) \right)
   \]

   Under the condition that \( \sum_{i=1}^{N} \pi_{i} = 1 \) , \( N \) is the number of knowledge node state, which means knowing and not knowing. Introducing Lagrange multiplier in Eq(6). \( \pi_{i} \) can be inferred as:

   \[
   \pi_{i} = \frac{P(O, i_{n} = | \tilde{\lambda})}{P(O | \tilde{\lambda})}
   \]

   In the same way, other parameters could be inferred as:

   \[
   a_{i_{n}} = \frac{\sum_{i_{n+1}} P(O, i_{n} = false | \lambda) \frac{\partial J(\alpha, \beta)}{\partial \alpha}}{\sum_{i_{n+1}} P(O, i_{n} = false | \tilde{\lambda})}
   \]

   \[
   \beta = \frac{\sum_{i_{n+1}} P(O, i_{n} = false | \lambda) \frac{\partial J(\alpha, \beta)}{\partial \beta}}{\sum_{i_{n+1}} P(O, i_{n} = false | \tilde{\lambda})}
   \]

   \[
   \delta_{i} = \frac{\sum_{i_{n+1}} P(O, i_{n} = false | \lambda) \frac{\partial L(\delta, \gamma)}{\partial \delta}}{\sum_{i_{n+1}} P(O, i_{n} = true | \lambda)}
   \]

   \[
   \gamma = \frac{\sum_{i_{n+1}} P(O, i_{n} = false | \lambda) \frac{\partial L(\delta, \gamma)}{\partial \gamma}}{\sum_{i_{n+1}} P(O, i_{n} = true | \lambda)}
   \]

   While \( J(\alpha, \beta) \) and \( L(\delta, \gamma) \) are guessing and slipping probability we just discussed.

   \[
   J(\alpha, \beta) = \frac{1}{1 + \exp[-\sum_{n}^{N} \alpha_{n} s_{n}^{(i)} + \beta]}
   \]

   \[
   L(\delta, \gamma) = \frac{1}{1 + \exp[-\sum_{n}^{N} \delta_{n} s_{n}^{(i)} + \gamma]}
   \]
The definition of \( S^{(n)} \), \( K_n \) and \( C_n \) are the same as KTLR-GS. However, \( P(L_0) \), \( P(T) \), \( P(F) \), \( P(G) \) and \( P(S) \) are different. The way we calculate \( P(L_0) \), \( P(T) \), \( P(F) \), \( P(G) \) and \( P(S) \) is:

- \( P(L_0) \): We extend the prior probability as logic regression of multiple subskills required in this learning process. The \( P(L_0) \) in this model can be calculated as follows:

\[
P(L_0 = \text{true}) = \frac{1}{1 + \exp\left(\sum_{i=0}^{m_n} \alpha_i s_i^{(0)} + \beta\right)}
\]

Here, \( \alpha_i \) represents skill \( i \)'s contributions to the prior probability, \( \beta \) is the offset of the prior probability, and \( s_i^{(0)} \) is an indicator of whether this skill is used in step 0.

- \( P(T) \): \( P(T) \) is the transition probability. With different subskills, it has different values in different steps. The learning probability in step \( n \) can be inferred as:

\[
P(K_{n+1} = \text{true}|K_n = \text{false}) = \frac{1}{1 + \exp\left(\sum_{i=0}^{m_n} \alpha_i s_i^{(0)} + \gamma\right)}
\]

Here, \( \alpha_i \) represents skill \( i \)'s contributions to the transition probability, \( \gamma \) is the offset, and \( s_i^{(0)} \) indicates whether the skill \( j \) is used in step \( n \). Since \( \alpha_i \) and \( \beta \) models the overall knowledge state of the student, they do not change over steps.

- \( P(F) \): \( P(F) \) is the forgetting probability. Just as the transition probability, it changes with different subskills in different steps. The forgetting probability in step \( n \) can be inferred as:

\[
P(K_{n+1} = \text{false}|K_n = \text{true}) = \frac{\exp\left(\sum_{i=0}^{m_n} \delta_i s_i^{(i)} + \gamma\right)}{1 + \exp\left(\sum_{i=0}^{m_n} \delta_i s_i^{(i)} + \gamma\right)}
\]

Here, \( \delta_i \) represents skill \( i \)'s contributions to the forget probability, \( \gamma \) is the offset.

- \( P(G) \) and \( P(S) \): \( P(G) \) is the guessing probability, and \( P(S) \) is the slipping probability, they can be affected by the difficulty of exercise items.

For example, if we have 10 problems, difficulty ranges from 1 to 5, we have 5 guessing probabilities and 5 slipping probabilities for students to solve each problem. Guessing and slipping probability can be inferred as:

\[
P(G) = P(C_n = \text{true}|K_n = \text{false}, I_n = m), m = 1, 2, 3, 4, 5
\]
\[ P(S) = P(C_n = \text{false}|K_n = \text{true}, I_n = m), m = 1, 2, 3, 4, 5 \]

The process of parameter estimating and performance predicting of KTLR-LFID are similar to the procedure in KTLR-GS.

The way to model item difficulty is shown as follows:

1) **Item difficulty Modeling:** There are lots of methods to model item difficulties. In this paper we use Item Response Theory [17] to model item difficulties. Item Response Theory is a theory of testing based on the relationship between individual’s performance on a test item and test takers’ level performance on an overall measure of ability that item was designed to measure. It uses the item response function to estimate the probability that a person with a given ability level will answer a question with a given difficulty correctly. With different amount of parameters in the item response function could be categorized into three types: 1PL (one parameter logistic model), 2PL and 3PL. In this paper we chose 2PL model to estimate the item difficulty. The item response function for 2PL model is:

\[
P_i(\theta) = \frac{1}{1 + \exp(-a_i * (\theta - b_i))}
\]

The parameters are interpreted as follows:

- \(a_i\): the discrimination of problem i.
- \(b_i\): the difficulty of problem i.
- \(\theta\): the test taker’s ability.

In order to introduce item difficulty into our model, we apply the IRT model on the answering sequence of all students, calculate the problem difficulty, normalize the item difficulty. The P(G) and P(S) could be expressed as:

\[
P(G_n) = P(C_n = \text{true}|K_n = \text{false}, b_n)
\]
\[
P(F_n) = P(C_n = \text{false}|K_n = \text{true}, b_n)
\]

\(b_n\) indicates the difficulty of item n we modeled in IRT model. \(P(G_n)\) means the guessing parameter in step n. \(P(F_n)\) means the forgetting parameter in step n.

C. **Extending Both Four Parameters In Knowledge Tracing Model (KTLR-FP)**

KTLR-GS and KTLR-LFID extend the transition parameters P(T), P(F) and emission parameters P(G), P(S) separately. Does it make sense to extend all the four parameters in the KT model to further enhance its effectiveness of describing multiple subskills in learning process? Under this assumption we propose KTLR-FP that extend all the four parameters in the KT model using logistic regression. The definition of the model is shown in Figure 5. The meanings of \(S_{(n)}\), \(K_n\) and \(C_n\) are the same as the other two models. The transition and emission parameters are a little different.

- \(P(L_0)\): it is defined in the same way as defined in KTLR-LFID. When a student start answering a set of questions the probability that he already knows the answer is:

\[
P(L_0 = \text{true}) = \frac{1}{1 + \exp(\sum_{i=0}^{n} a_i * s_i^{(0)} + \beta_0)}
\]

- \(P(T)\): The transition probability in step n can be calculated as:

\[
P(K_{n+1} = \text{false}|K_n = \text{true}) = \frac{\exp(\sum_{i=0}^{m} \delta_i * s_i^{(0)} + \gamma)}{1 + \exp(\sum_{i=0}^{m} \delta_i * s_i^{(0)} + \gamma)}
\]

- \(P(F)\): The forgetting probability can be inferred as:

\[
P(K_{n+1} = \text{false}|K_n = \text{true}) = \frac{\exp(\sum_{i=0}^{m} \nu_i * s_i^{(1)} + \epsilon)}{1 + \exp(\sum_{i=0}^{m} \nu_i * s_i^{(1)} + \epsilon)}
\]

- \(P(G)\): Just as we describe in model KTLR-GS, the guessing probability can be inferred as:

\[
P(C_n = \text{true}|K_n = \text{false}) = \frac{1}{1 + \exp(\sum_{i=0}^{m} \theta_i * s_i^{(0)} + \lambda)}
\]

- \(P(S)\): The slipping probability can be inferred as:
\[
P(C_n = \text{false}|K_n = \text{true}) = \frac{\exp\left(\sum_{i=0}^{m} \mu_i \cdot s_n^{(i)} + \omega\right)}{1 + \exp\left(\sum_{i=0}^{m} \mu_i \cdot s_n^{(i)} + \omega\right)}
\]

In this model, we use logistic regression to express the learning, forgetting, guessing, slipping and prior probability. This means, we will estimate more parameters than last two models. So compared with other two models, it’s reasonable to speculate this model is more likely to be effected by random errors happen during the estimation.

IV. DATASETS

We evaluate KTLR-GS, KTLR-LFID, and KTLR-FP on an open dataset ASSISIments dataset and a private dataset collected through our own learning record system.

A. The ASSISIments Dataset

Our first open dataset comes from ASSISTments [18], a web-based math tutoring platform. It is best known for its 4th - 12th grade math content. In the 2004-2005 school year some 600+ students use this platform for about two weeks. If students got the item correct they were allowed to continue with a new question. If they got it wrong, they were provided with a small “tutoring” session where the problem is broken down into steps for student to get it correct step by step. In our original datasets, there are 912 students, 226100 actions. The longest student answering sequence is 768 the shortest is 1.

In order to fit our model, we chose students who finished more than 20 problems. In the end our actual dataset consists 797 students, 209500 records. Performance records of each student were logged for 106 skills (adding-decimals, addition, division etc.). The overall correct percent is 54.41%.

B. Students Algebra Dataset

The private dataset was collected from our learning record system storing 14,037,146 learning behavior data from 140 schools and 9 online educational companies. We extracted student performance data of a math class from five junior middle schools.

There were five chapters in this class, where teacher assigned twenty-five questions. Students finished the homework through a math tutoring system, after that the tutoring system sent the student performing data to our learning analysis platform in xAPI [19], a brand new specification for learning technology to store and share learning data across the platform. Each student had three chances to answer a question, when a student attempted a question for the third time, he was allowed to view the hint. Our system records the student’s performance on each attempt, but we only mark the question correctly if the student answers it correctly on the first attempt.

In the raw dataset, there are totally 262,791 actions. After filtering the answers after the first attempt, we obtained 152,500 actions. The overall correct rate of the questions is 73.27%. The student number in each section ranges from 972 to 1578, and all of the student finished 25 questions. Some of them finished the whole five chapters, while some only finished part of chapters.

V. EXPERIMENT

We implement the models in the Bayes Net Toolbox for Matlab (BNT) [20], an open source toolbox for Bayesian network constructing. Specifically, we define the knowledge node K given n subskills Sj as a “softmax” node in the toolbox. To evaluate our models, we fit them separately for each student on the first 80% of all of that student’s data, test on the second 20%, and average the test results across students. We repeat the experiment for 10 times and average the result. For comparison, we also apply the LR-DBN, KT-IDEM and KT model and estimate the student’s performance on the second 20% data. The criteria we evaluate the performance of the models are listed as follows:

- MAE: Mean Absolute Error
- RMSE: Root Mean Squared Error
- MSE: Mean Squared Error
- AUC: Area Under the Curve

A. The ASSISIments Dataset

We apply model KTLR-GS, KTLR-FP, LR-DBN and KT on the ASSISIments dataset. The KTLR-LFID model cannot be applied on this dataset because we notice that it’s difficult to find an appropriate sequence that the majority of students have answered to evaluate the item difficulty through IRT model. In the same reason, we didn’t apply KT-IDEM on this dataset.

The result of the models is shown in Table I. KTLR-GS and KTLR-FP perform better than other models. The AUC of KTLR-FP is 0.095 higher than LR-DBN, while KTLR-GS has 0.1315 higher AUC than LR-DBN. The results from evaluating the models are strongly in favor of KTLR-GS, with lower MAE, EMSE, MSE and higher AUC.

LR-DBN only considers transition parameters, KTLR-FP not only extends transition parameters but also extends emission parameters. Just as discussed in Algorithm 1, predicting state of knowledge and performance nodes with guessing and slipping probability. So extending guessing and slipping probability helps improving the model.

KTLR-GS uses logistic regression to update guessing and slipping parameters in the model. KTLR-GS reveals a better performance than LR-DBN, so we conclude that extending guessing and slipping parameters improves the model more significantly than extending learning and forgetting parameters. In [13] Bremes extends KT model using general student and item features on guessing and slipping parameters, leads to the same conclusion.

Table I Results of models on the ASSISIments platform dataset

<table>
<thead>
<tr>
<th></th>
<th>MAE</th>
<th>RMSE</th>
<th>MSE</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>KTLR-GS</td>
<td>0.2843</td>
<td>0.3558</td>
<td>0.1266</td>
<td>0.8578</td>
</tr>
<tr>
<td>KTLR-FP</td>
<td>0.3318</td>
<td>0.423</td>
<td>0.1789</td>
<td>0.8213</td>
</tr>
<tr>
<td>LR-DBN</td>
<td>0.3264</td>
<td>0.4398</td>
<td>0.1934</td>
<td>0.7263</td>
</tr>
<tr>
<td>KT</td>
<td>0.3856</td>
<td>0.4752</td>
<td>0.2258</td>
<td>0.7069</td>
</tr>
</tbody>
</table>

B. Student Algebra Dataset

The answering pattern in algebra dataset is well ordered. So we apply KTLR-GS, KTLR-LFID, KTLR-FP, LR-DBN, KT-IDEM and KT on this dataset. There are five chapters in
this dataset, we model each chapter separately, each chapter we apply the model for 10 times and average the performance.

Unlike the ASSISments dataset, problems have the subskills labeled for each step, what we have about this dataset is the problem description and answering outcomes. How to label the subskill for each question? We invited two experienced junior school teachers to help us annotating the subskills. They suggested us to separate the inverse function into six subskills, which are definition of inverse function, image of inverse function, expression of inverse function, property of inverse function, geometric application of inverse function and synthesis of inverse function. We use the CFM to estimate the effectiveness of the division of the subskill.

1) Why multiple subskills is more efficient than single skill: CFM(conjunctive factor model) [21] is a popular model in LFA(Learning Factor Analysis) [22], which is a framework aiming to address the problem of discovering a better cognitive model from student learning data. CFM assumes that when an item requires multiple skills present, the item is harder than the items requiring only one of those skills. Using the CFM model to fit the answering sequence, the cognitive model with lower errors is the candidates of being better cognitive models.

In the traditional KT model, each question has the same subskill, which means that there is only one skill. However, in our model in we assume there are six subskills, which assumption is better? We use CFM model to confirm this question. The overall results of single skill and six subskills are shown in Table II.

<table>
<thead>
<tr>
<th></th>
<th>Single Skill</th>
<th>Six Subskills</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIC</td>
<td>3404.352</td>
<td>3356.419</td>
</tr>
<tr>
<td>BIC</td>
<td>5525.388</td>
<td>5525.149</td>
</tr>
</tbody>
</table>

AIC and BIC are measure of the relative quality of statistical models for a given set of data, the smaller the AIC and BIC are, the better the model we estimate is. It shows that conjunctive model with six subskills performs better than single skill.

2) Results: We apply the models on each section, then average the results of five sections, the overall results are shown in Table III. As shown in Table III, both three models perform better than LR-DBN, KT-IDEM and KT. The AUC of KTLR-GS is 0.0742 higher than LR-DBN, the AUC of KTLR-LFID is 0.1039 higher than LR-DBN, and KTLR-FP has a 0.1673 higher AUC than LR-DBN. KTLR-FP preforms better than any other models on this dataset. KTLR-LFID performs better than KTLR-GS with lower MAE, RMSE, MSE and higher AUC.

<table>
<thead>
<tr>
<th></th>
<th>MAE</th>
<th>RMSE</th>
<th>MSE</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>KTLR-GS</td>
<td>0.3991</td>
<td>0.4685</td>
<td>0.2195</td>
<td>0.7998</td>
</tr>
<tr>
<td>KTLR-LFID</td>
<td>0.2792</td>
<td>0.4244</td>
<td>0.1863</td>
<td>0.8295</td>
</tr>
<tr>
<td>KTLR-FP</td>
<td>0.3073</td>
<td>0.3627</td>
<td>0.1331</td>
<td>0.8929</td>
</tr>
<tr>
<td>LR-DBN</td>
<td>0.3541</td>
<td>0.4725</td>
<td>0.2233</td>
<td>0.7256</td>
</tr>
<tr>
<td>KT-IDEM</td>
<td>0.4007</td>
<td>0.4741</td>
<td>0.2248</td>
<td>0.7134</td>
</tr>
<tr>
<td>KT</td>
<td>0.4139</td>
<td>0.4747</td>
<td>0.2254</td>
<td>0.7041</td>
</tr>
</tbody>
</table>

C. Discussion

In this section, we present the experimental results of the models on each dataset. The results confirm that all the models including KTLR-FP, KTLR-GS and KTLR-LFID performs better than LR-DBN, KT-IDEM and KT. KTLR-FP underperforms KTLR-GS on Assisments dataset, but outperforms KTLR-GS on algebra dataset. KTLR-LFID outperforms KTLR-FP on algebra dataset. It is necessary to investigate why KTLR-FP underperforms KTLR-GS on Assisments dataset, but outperforms KTLR-GS on algebra dataset.

From the beginning, we extracted the error cases in the predictions made by both models and examined the sequence length of each student for whom our model fails to predict performance correctly. The histogram of sequence lengths of error predictions can be plotted in Figure 6.a and Figure 6.b.

The range of error sequence of two models is similar, which means that the capacity of two models resembles each other. The distribution of error sequence length is mainly located in the range of 100 to 500, because the shorter sequence can accelerate the convergence of EM-based parameter fitting. Also with longer sequences, the models have more information of student performance, thus leading to a higher probability to make correct predicts. We conclude that with the sequence range in 50 to 500, strictly speaking 27 to 628, KTLR-FP performs unstably.
Then we checked the procedure of parameter estimation and performance prediction. After comparing the transition and emission parameters trained in both correct prediction and error prediction cases, we find the transition and emission parameters of most error predictions apparently deviate from normal value ranges, which are either too large or too small. Figure 7 shows such a case where guessing and slipping probability of an error prediction is extremely smaller than those of a correct prediction along their answering sequence.

![Figure 7 Guessing and slipping probability of correct and error predictions](image)

Figure 7 Guessing and slipping probability of correct and error predictions

We estimate the coefficient of transition and emission parameters based on Eq(9-12), but we couldn’t find the correct gradient for $\alpha_j$ (the coefficient of subskill j), if it is not involved in a specific step. For $S_j^{(n)} = 0$ in step n, then Eq(1) and Eq(2) do not contain coefficient $\alpha_j$. Under this condition, we couldn’t estimate $\alpha_j$. So it’s possible that we will misestimate the coefficient of subskills that are not used in most steps of the answering sequence. With the most variables between the models, KTLR-FP will magnify the mistake during the parameter estimation. So we speculate that KTLR-FP will make more mistake with the sparse subskill matrix than the dense matrix.

Figure 8.a and Figure 8.b display the subskill distribution of the ASSIStments dataset and algebra dataset respectively. The ASSIStments subskill matrix seems to be a sparse matrix where there are so many subskills but only very few of them are involved in a specific step. In contrast, the subskill matrix of algebra dataset seems to be much denser. Thus KTLR-FP makes more mistakes on the ASSIStments dataset.

In this way we conclude that EM process of parameter estimation is likely to make mistakes when the subskill matrix is a sparse matrix.

Based on the examination we carried out, we conclude that KTLR-FP underperforms KTLR-GS when the answering sequence is between 27 and 628, or the subskill matrix is a sparse matrix. On the condition that student answer sequence is under 27 or over 628, or the subskill matrix of the problems is dense matrix, KTLR-FP outperforms KTLR-GS.

![Figure 8 Part of the step x subskill matrix of the ASSIStments dataset and algebra dataset. Blue means subskill j is not involved in step n, red means step n inspects subskill j.](image)

(a). Part of the step x subskill matrix of the ASSIStments dataset, x-axis represents the subskills, y-axis represents the steps.

(b). Part of the step x subskill matrix of the algebra dataset, x-axis represents the steps, y-axis represents the subskills.

VI. CONCLUSION

In this paper we introduce Knowledge Tracing Model, and three groups of extensions in multiple subskills modeling. They are conjunctive models, performance factor analysis and logistic regression based models. However, neither of the models has fully combined the transition and emission parameters in KT model with different subskills.

This paper extends KT model with logistic regression within a dynamic Bayesian network by proposing three new models KTLR-GS, KTLR-LFID and KTLR-FP. They extend different parameters in KT model, and explore the effect of extension of different parameters. The models were implemented with a widely used Bayesian network toolbox BNT.

To evaluate our models, we use two datasets: ASSIStments dataset, a very famous open dataset and student algebra dataset collected from our learning record system. The models are compared with several published models including LR-DBN, KT-IDEM and KT. Overall, the experimental results on both datasets demonstrate that all the models proposed in the paper (KTLR-GS, KTLR-LFID and KTLR-FP) perform better than LR-DBN, KT-IDEM and KT. Furthermore, we conduct the detailed analysis of the EM based parameter estimation process for our models to investigate the different performance for the model KTLR-GS, and KTLR-FP under the two datasets. Interestingly, we observe that the subskill association matrix in the steps of student problem solving has interesting impact on the parameter estimation of these models. Preliminary results indicate that the KTLR-GS model is more suitable in the scenario of the sparse subskill association matrix whereas the KTLR-FP seems more suitable in the dense scenario.

ACKNOWLEDGMENT
This work was supported by grant from NSFC (Grant No. 61532004) and State Key Laboratory of Software Development Environment (Funding No. SKLSDE-2017ZX-04).

REFERENCES