Journal of Machine Learning Research x (2010) x-xx Submitted KDD Cup 2010; Published xx/xx

# Knowledge Tracing with Stochastic Method

**Tri Kurniawan Wijaya** TRIKURNIAWANWIJAYA@GMAIL.COM
*Department of Computer Science*
*Technische Universitat Dresden*
*Dresden 01069, Germany*

**Philips Kokoh Prasetyo** PHILIPS@COMP.NUS.EDU.SG
*School of Computing*
*National University of Singapore*
*Singapore*

**Editor:**

## Abstract

This paper describes our approach for KDD Cup 2010. The approach we used is to utilize a stochastic optimization in knowledge tracing and also introduce a new indicator variable, "luck". In order to compute the general probability about student's understanding of a particular problem, we use exponential moving average to give more weighting to the recent results. In addition, we also taking into account several strategies to combine several knowledge components in parallel.

**Keywords:** stochastic method, knowledge tracing, exponential moving average, kdd cup 2010

## 1. Introduction

The challenge of KDD Cup 2010 is to predict student performance on mathematical problems from logs of student interaction with Intelligent Tutoring Systems. The data contain several columns which include student id, problem hierarchy, problem name, step name, knowledge needed to answer the problem and several indicators about whether the students correctly answer the problem or not. The goal of KDD Cup 2010 is to predict Correct First Attempt from the given dataset. In our approach, we will consider the student id, problem hierarchy, problem name, knowledge components and opportunity. A knowledge tracing approach is employed to solve the problem. Corbett and Anderson (1995) shows that knowledge tracing can be used to model knowledge state during skill acquisition.

## 2. The Method

This section will describe about our approach to solve the problem. First we will explain how we deal with the dataset and preprocessing. Next we describe the basic formula that we used in our method. Our main contribution that is stochastic optimization in knowledge tracing is explained in the next subsection. The last subsection describes how we apply our model to the test dataset.

### 2.1 Proprocessing

In order to be able to apply knowledge tracing, a preprocessing to reconstruct data sequence is previously carried out. In this step, for every student, the rows from test data is placed below the rows of training data based on corresponding unit in problem hierarchy. Below is the example:

| Rows | Student Id | Unit | Dataset |
|---|---|---|---|
| 1-100 | Student1 | Unit1 | Training |
| 101-120 | Student1 | Unit1 | Test |
| 120-221 | Student1 | Unit2 | Training |
| 222-236 | Student1 | Unit2 | Test |
| ... | ... | ... | ... |

However, in some cases, sequence cannot be reconstructed due to separated sequence of a unit. For this case, we just simply put the rows from the test data after the last segment, for example:

| Rows | Student Id | Unit | Dataset |
|---|---|---|---|
| 1-100 | Student1 | Unit1 | Training |
| 101-120 | Student1 | Unit2 | Training |
| 120-221 | Student1 | Unit1 | Training |
| 222-236 | Student1 | Unit1 | Test |
| ... | ... | ... | ... |

### 2.2 Basic Formula

After data sequence is reconstructed, we apply the following formula to determine the probability a student correctly answer the questions in the first attempt ($Q = correct$).

$$P(Q = Correct)new = (1 - P(S = know)) * guess + P(S = know) * (1 - slip) + luck \quad (1)$$

Formula in eq. 1 is a modification from the formula in eq. 2 (Pardos et al., 2008). We add *luck* factor to the formula alongside guess and slip factor. $P(S = know)$ is the prior probability of known skills. This probability is computed from the average probability of each component of each student based on the knowledge he/she has got before.

$$P(Q = Correct) = (1 - P(S = know)) * guess + P(S = know) * (1 - slip) \quad (2)$$

Since we would like to give a more weight to the most recent result, exponential moving average (Cox, 1961) is utilized to compute the average. Dealing with data which have more than one knowledge components (KCs), we employ four strategies to combine different knowledge components, i.e. mean (a normal average of the average probability of each KCs), multiply (multiply the average probability of all KCs), max (the maximum value of the average probability of the KCs) and min (the minimum value of the average probability of the KCs). After this operation, we have four types of $P(S = know)$. For each row in the test data, the probability $P(S = know)$ based on four strategies above.

### 2.3 Stochastic Optimization

First, we would like to exploit the similarity behavior that could be hidden in the data. In this case, we created 4 types of separation of the data, based on student id, problem hierarchy, problem name, and the number of the knowledge components. Separation by student id yields around 3000 groups for algebra_2008_2009 dataset and 6000 groups for bridge_to_algebra_2008_2009. Separation by problem hierarchy, problem name, and the number of knowledge components yields about 160, 180000, and 12 groups for algebra_2008_2009 dataset and 180, 50000, and 14 groups for bridge_to_algebra_2008_2009 dataset respectively.

For each separation and each group, we generate $n$ random number for *guess* and $n$ random number for *slip* separately. Based on the empirical result, we pick 100 as the $n$, although assigning lower (around 50) or higher value (around 200) could be possible. Next, we calculate $P(Q = correct)$ for each data using the *guess* and *slip* we define before (from the random number list) and we try the four combination strategies as we define before (mean, multiply, max and min). In total, we have to do $4 * n$ calculation. $Rmse$ is then calculated for each group. The best configuration (combination strategy, guess, and slip value), i.e. the configuration which yield the smallest $rmse$ is saved for each group.

Finally, in order to determine *luck*, we apply the algorithm above in similar manner. For each separation, we generate a list of $n$ random number for each group. Then, we calculate $P(Q = correct)new$ for each random number we generate. In total, we process $n$ calculation. For each calculation, we also calculate the $rmse$. The value which yields the smallest $rmse$ is preserved as the best "luck" configuration for the group. In addition we also save the smallest $rmse$ that we can get for the group.

### 2.4 Applying the Model

The configuration that we have found in the previous section is the key for our model. Please note that our configuration for each group in each separation includes the guess, slip, combination strategy, luck and rmse for each group for each separation. In the test dataset, we have to recognize the group for each separation for each row (basically we have to recognize the value of its student id, problem hierarchy, problem name, and the number of the knowledge components). For example:

| Rows | Student Id | Problem Hierarchy | Problem Name | Knowledge Component |
|---|---|---|---|---|
| 101 | Student1 | Unit 1, Section 1 | Problem1 | Addition, Equality |
| ... | ... | ... | ... | ... |

The data above would lie to the group "Student1" on the separation student id, group "Unit 1, Section 1" on the separation problem hierarchy, group "Problem1" on the separation problem name, and group "2" on the separation the number of the knowledge components.

For the next step, we look at the each group that we have found. We have 4 groups, one group for each separation. Choose the group (in above example, look at the group "Student1", "Unit1, Section 1", "Problem1", and "2") that has the smallest $rmse$. Then use the *guess*, *slip*, *luck* and combination strategy to calculate $P(Q = correct)new$. The result is the answer for $CorrectFirstAttempt$ for the corresponding row.

## 3. Conclusions

We have described our knowledge tracing with stochastic method for the KDD Cup 2010. We found that our simple approach performs reasonably well for the problem. One of the important steps for this approach is the preprocessing to reconstruct data sequence. We believe that if the data sequence can be reconstructed well, this approach will achieve better results. A possible extension to improve this approach is feature weighting which give importance notion of certain knowledge. Finally, we feel that this task gives us some experiences in mining data for educational purposes.

Thank you to the organizers and sponsors for hosting KDD Cup 2010.

## References

Albert T. Corbett and John R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1995.

David R. Cox. Prediction by exponentially weighted moving averages and related methods. *Journal of the Royal Statistical Society. Series B (Methodological)*, 23(2):414–422, 1961.

Zachary A. Pardos, Joseph E. Beck, Carolina Ruiz, and Neil T. Heffernan. The composition effect: Conjunctive or compensatory? an analysis of multi-skill math questions in its. *In EDM'08: Proceedings of 1st International Conference on Educational Data Mining*, 2008.

## Appendix A.

This appendix explains the exponential moving average. The formula of exponential moving average is defined as follows:

$$S(t) = \alpha * Y(t) + (1 - \alpha) * S(t - 1)$$

Where $S(t)$ is the average of time $t$, $\alpha$ is a real number between 0 - 1 (in this case, we use 0.1), $Y(t)$ is the observation of time $t$. Exponential moving average is not necessarily use since $t = 1$. For several $t$ in the beginning, we shall use a normal average. In our approach, we use the normal average calculation until $t = 6$ . We use exponential average from $t = 7$. For example:

- The probability knowledge of 1010 is 0.5 (we use the normal average calculation)
- The probability knowledge of: 10101111 is 0.73. For the first six data (101011), $S(6)$ is 0.6667. Then $S(7) = 0.1*1+(1-0.1)*0.6667 = 0.7$, and $S(8) = 0.1+1+(1-0.1)*0.7 = 0.73$.