Topological Order Discovery via Deep Knowledge Tracing

Jiani Zhang[1,2](✉) and Irwin King[1,2]

[1] Shenzhen Key Laboratory of Rich Media Big Data Analytics and Applications, Shenzhen Research Institute, The Chinese University of Hong Kong, Shenzhen, China
{jnzhang,king}@cse.cuhk.edu.hk

[2] Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong

Abstract. The goal of discovering topological order of skills is to generate a sequence of skills satisfying all prerequisite requirements. Very few previous studies have examined this task from knowledge tracing perspective. In this paper, we introduce a new task of discovering topological order of skills using students' exercise performance and explore the utility of Deep Knowledge Tracing (DKT) to solve this task. The learned topological results can be used to improve students' learning efficiency by providing students with personalized learning paths and predicting students' future exercise performance. Experimental results demonstrate that our method is effective to generate reasonable topological order of skills.

Keywords: Knowledge tracing · Topological order · Recurrent neural networks

1 Introduction

Online education platforms have gained great popularity in recent years. Companies like Coursera and edX have attracted millions of students to enroll diversified online courses. However, these Massive Open Online Courses (MOOCs) are contributed by different institutions without an integrated structure. Moreover, in these online environments students often lack personalized instruction that helps them study more efficiently.

In order to give personalized instruction, we need to evaluate what a student knows and does not know in advance. Knowledge Tracing (KT) is such a task of modeling students' latent skills over time based on past study performance, where study performance is a sequence of exercises with correct or incorrect responses. Usually, we model students' mastery of skills as latent variables and students' performance as observed variables. As shown in Fig. 1, each exercise requires an underlying skill to answer the exercise correctly. Different exercises can map to the same skill, e.g., $Ex.1$ and $Ex.2$ both map to $SkillA$. If a student answers an exercise correctly, then the probability of his mastery of the

© Springer International Publishing AG 2016

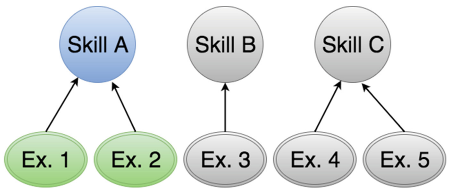


Fig. 1. Sequence of one student's exercise performance. Green nodes represent exercises with correct answers, while gray nodes represent exercises with incorrect answers. Blue skills are in *known* state and gray ones are in *unknown* state. (Color figure online)

underlying skill will increase. Otherwise, the probability will decrease. The most promising application of accurately evaluating students' underlying skills is to help determine which exercise is most suitable to give to students.

However, in previous KT studies, underlying skills are treated as isolated and independent individuals, which is unrealistic in reality. For example, when learning arithmetic operations, a student would not master *subtraction* before this student mastered *addition*. Among skills, there always exists a topological order which is an optimal sequence for students to learn skills one after another. As shown in Fig. 2, $SkillB$ is a prerequisite of $SkillC$. If a student has not mastered $SkillB$, then the probability of correctly answering $Ex.4$, whose underlying skill is $SkillC$, will decrease. This observation makes it possible to discover topological order of skills from students' exercise performance. The challenge of this task is that (1) the input data is only a sequence of binary responses to student's answers, (2) it is inherently difficult to represent human learning process by numerical simulations, and (3) there is no explicit description of skills in the observation data.

In this paper, we propose a rule-based method to discover topological order of skills from limited students' performance data with the aid of Deep Knowledge Tracing (DKT). DKT is a newly proposed method [1] to solve the KT problem using Recurrent Neural Networks (RNNs). One big advantage of deep learning methods is their capability of learning feature representation from large-scale datasets, where domain knowledge and structure can be discarded [2]. Since it

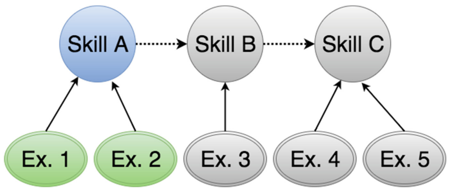


Fig. 2. Adding topological order of skills to predict students' future performance. For example, directed edge (A, B) indicates that $SkillA$ must be completed before $SkillB$ may be attempted and the same as directed edge (B, C).

is difficult to generate topological order of skills directly from students' performance data, we can use the order of students' mastering of skills as a bridge. Our method infers students' mastery of skills from students' exercise performance, and then discovers prerequisite skill pairs from the order of students' mastery of skills, and finally generates the topological order of all skills. Experimental results demonstrate the effectiveness of our approach.

We summarize three main contributions in our paper.

1. This is the first paper to introduce the task of discovering topological order of skills from students' exercise performance.
2. We propose a new method to discover topological order of skills utilizing Deep Knowledge Tracing.
3. Experimental results demonstrate the effectiveness of our method.

2 Related Work

Knowledge Tracing (KT) has increasingly received attention both in the psychology and computer science domain in the past two decades.

The dominant method of KT is Bayesian Knowledge Tracing (BKT), which was first introduced in 1994 [3] to implement mastery learning. Mastery learning maintains that a level of mastery must be achieved in prerequisite knowledge before proceeding to learn subsequent topics. BKT can be easily described with two learning parameters and two performance parameters. Many following variations raised by integrating personalization study [4,5], exercise diversity [6] and other information into Bayesian framework.

The BKT model and its extensions are the most popular models in Intelligent Tutoring Systems due to their strong interpretation properties of evaluating students latent knowledge state. However, several strong assumptions proposed in the first paper [3] have not improved yet in the follow-up study. Assumptions such as bnginary knowledge state representation, no forgetting mechanism and single skill modeling are all the causes that make BKT inflexibility and unrealistic.

Recently, Chris Piech et al. proposed a Deep Knowledge Tracing (DKT) [1] method that used Recurrent Neural Networks (RNNs) [7,8] to trace student's knowledge, which achieved great improvement on the prediction accuracy of students' performance over previous models.

Deep Learning has achieved great success in pattern recognition and machine learning domains [2,9], such as computer vision, natural language processing and speech recognition. However, deep neural networks have not attracted much attention in educational data mining. DKT was the first model to integrate deep learning models into knowledge tracing. The DKT model implemented the simplest one-hidden-layer RNNs, but it demonstrated a stunning improvement over the mainstay, i.e., BKT [10]. The prediction accuracy of students' future performance increased more than 15 %.

3 Topological Order Discovery Model

The goal of topological order discovery can be achieved by two main steps. The first step is to obtain students' mastery of skills from their exercise performance using DKT. The second step is to discover topological relationship between skills from students' mastery order of skills by two predefined rules.

3.1 Deep Knowledge Tracing Model

The DKT model [1] was primarily designed to predict the probability of answering the next exercise correctly. However, the actual output of this model is the probability of answering all the exercises correctly. From student's current exercise performance, we can easily find the exercise with the highest or lowest correct probability in the next time stamp within this model.

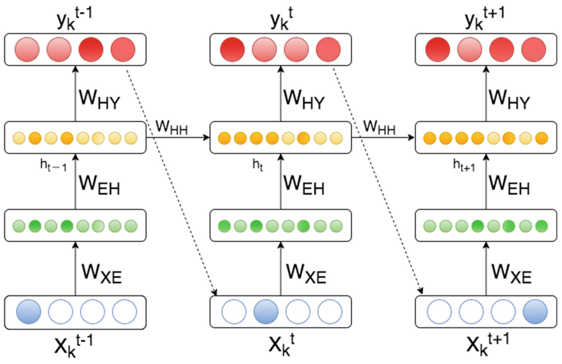


Fig. 3. Deep knowledge tracing model

As shown in Fig. 3, the framework for DKT is Recurrent Neural Networks (RNNs), where at each time t the input $\mathbf{x_k^t}$ from student k is an exercise tuple, i.e., $\mathbf{x_k^t} = \{q_k^t, a_k^t\}$. Exercises which require the same skill to answer correctly are labeled with the same label index in the preprocessing. Suppose there are N unique skills, the input $q_k^t \in \mathbb{R}^N$ is a one-hot encode presentation where only the corresponding index equals to 1 and others are all 0s. Accordingly, a_k^t is also $\in \mathbb{R}^N$ and preprocessed to one-hot encoding with the corresponding exercise index equals to 1 if answered correctly. The combined input for RNNs, i.e., $\mathbf{x_k^t}$, is $\in \mathbb{R}^{2N}$.

The output $\mathbf{y_k^t} \in \mathbb{R}^N$ is the probability that student k will answer each exercise correctly in the next time stamp. Evaluation function is the negative log likelihood. Let $\delta(q_k^{t+1})$ denote which exercise to be answered at time stamp $t+1$ and ℓ be cross entropy. The total loss for an exercise sequence is

$$\text{Loss} = \sum_t \ell((\mathbf{y_k^t})^T \delta(q_k^{t+1}), a_k^{t+1}) \tag{1}$$

Then we can use standard backpropagation algorithm [11] to train the model by computing gradient descents of the loss function.

3.2 Topological Order Discovery

After training the DKT model well, we can acquire the probability of answering all exercises correctly in the next time stamp. The relationship between two skills, e.g., skill A and B, can be divided into four different categories.

1. If B has been mastered, then A has also been mastered. $\Rightarrow$ 'master B $\rightarrow$ master A'.
2. If B has not been mastered, then A has not been mastered either. $\Rightarrow$ '$\neg$ master B $\rightarrow$ $\neg$ master A'.
3. If B has been mastered, but A has not been mastered yet. $\Rightarrow$ 'master B $\rightarrow$ $\neg$ master A'.
4. If B has not been mastered, but A has already been mastered. $\Rightarrow$ '$\neg$ master B $\rightarrow$ master A'.

Among these four cases, case 1 and case 2 can help generate prerequisite pairs. In other words, if the skill related to current exercise has been mastered, then the skill whose exercise has the highest probability to be answered correctly in the next time stamp can be regarded as a candidate prerequisite skill. Otherwise, the skill whose exercise has the lowest probability can be regarded as a candidate prerequisite skill.

Since each exercise is closely related to one underlying skill in the DKT model, we can use its skill label as its exercise label. Thus, the probability of answering an exercise correctly is the same as the probability of mastering its latent skill. We propose the following method to discover topological order of skills from students' exercise performance.

Firstly, generate partial order pairs. Since students always answer exercises with the same skill in sequence, we only need to care about whether this student has mastered the underlying skill at the last time stamp. If the probability of answering the input exercise is larger than 0.5, then it infers that the student has mastered this skill and vice versa. We apply two rules to generate partial order pairs: (1) if the current skill has been mastered, then the skill with the highest output probability can be regarded as a prerequisite, and (2) if the current skill has not been mastered, then the skill with the lowest output probability can be considered as a prerequisite.

Secondly, remove redundant links between skills. The most useful rule is that if '$skillA \rightarrow skillB$', '$skillB \rightarrow skillC$' and '$skillA \rightarrow skillC$', then '$skillA \rightarrow skillB \rightarrow skillC$' is enough. Pruning is an essential part for generating the final topological order of skills.

Last, use topological sorting algorithm to generate one topological order of skills. We apply Kahn's algorithm [12] to generate a topological order from the directed acyclic graph discovered in the previous step.

4 Experiments

In this section, we conduct experiments on real-world datasets to evaluate the effectiveness of our method.

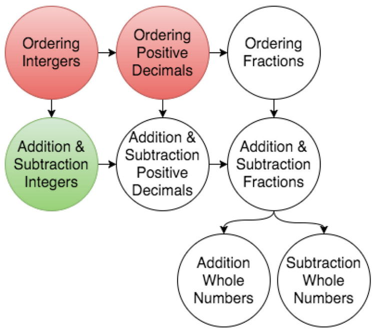
4.1 Datasets

ASSISTment Dataset: This dataset was gathered in year 2009–2010 from the ASSISTments platform.[1] There are two partitions, one with labeled skills and the other without labeled skills. In our experiments, we train the model using the 'skill builder' one, which is a large, standard benchmark in KT topic. In this dataset, there are more than 4,000 students having answered over 446,000 exercises along with 111 unique labeled skills. Two assumptions are imposed on this dataset: (1) each exercise maps to only one skill, and (2) exercises with the same skill are preprocessed to have the same exercise tags.

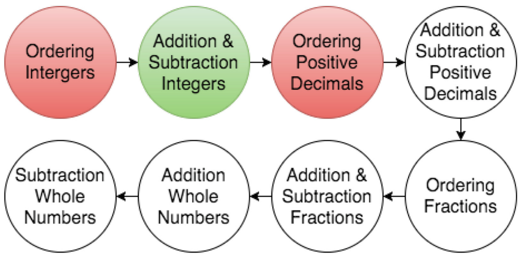
In order to evaluate the plausibility of extracted topological order of skills, a hierarchical skill graph and its topological order in arithmetic subject was created based on empirical knowledge. Figure 4 demonstrates the ground truth among eight example skills.

4.2 DKT Model Results

First we train the DKT model on the ASSISTment dataset until the AUC value attains 0.86. In the process of training DKT, all one-hot encoding exercises and


Ordering Intergers
Ordering Positive Decimals
Ordering Fractions
Addition & Subtraction Integers
Addition & Subtraction Positive Decimals
Addition & Subtraction Fractions
Addition Whole Numbers
Subtraction Whole Numbers

(a) Skill graph


Ordering Intergers
Addition & Subtraction Integers
Ordering Positive Decimals
Addition & Subtraction Positive Decimals
Ordering Fractions
Addition & Subtraction Fractions
Addition Whole Numbers
Subtraction Whole Numbers

(b) Topological Order of skills

Fig. 4. A partial manually-labeled skill graph and its topological order in Arithmetic subject from the ASSISTment dataset. Each node represents a specific skill. Each arrow represents prerequisite relationship. Red nodes denote that the student has mastered these skills and green nodes can be the next highly-recommended skills for this student to learn. (Color figure online)

[1] ASSISTment dataset: https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data.

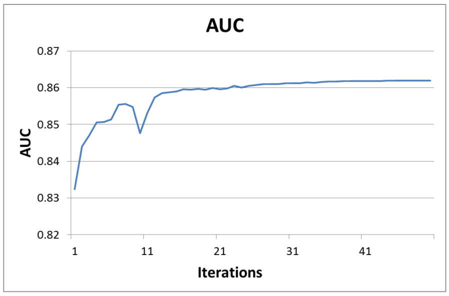


Fig. 5. The AUC in testing time over 50 iterations

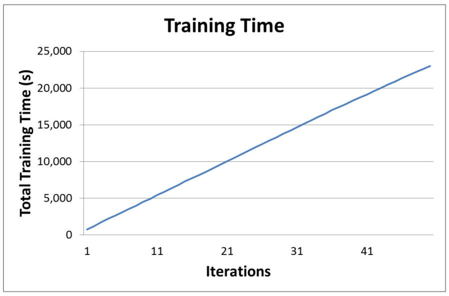


Fig. 6. Total training time over 50 iterations

responses are mapped to a fixed-size dense space. We tried several values of the space size and finally set it to be 100.

Furthermore, we consistently use hidden dimensionality of 100 and only one hidden layer. More hidden layers can instead lead to overfitting issue and decrease the AUC value.

Figure 5 gives the AUC value on the testing data within 50 iterations. We can see that it converges very fast and reaches 0.86 after the 23rd iteration. The time for training each iteration requires nearly 10 min, which is shown in Fig. 6.

4.3 Topological Order Discovery Result

The prediction accuracy on the ASSISTment dataset guarantees the feasibility of using the topological order discovery model to obtain recommended learning order of latent skills in the dataset. After replacing skill IDs with labeled skill names, the skill graph reveals an interpretable ordering of skills within a certain subject. Compared with the manually-described topological order of skills (see Fig. 4), we can see that our model can accurately discover the topological relationship between skills (see Fig. 7). It only fails to find the relationship between

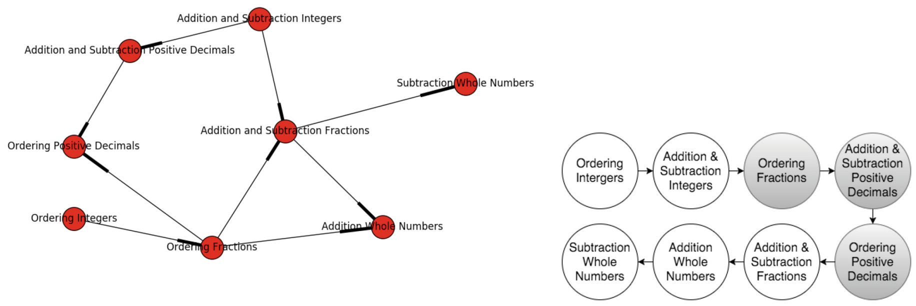

Addition and Subtraction Integers
Addition and Subtraction Positive Decimals
Subtraction Whole Numbers
Addition and Subtraction Fractions
Ordering Positive Decimals
Ordering Integers
Addition Whole Numbers
Ordering Fractions
Ordering Intergers
Addition & Subtraction Integers
Ordering Fractions
Addition & Subtraction Positive Decimals
Ordering Positive Decimals
Addition & Subtraction Fractions
Addition Whole Numbers
Subtraction Whole Numbers

Fig. 7. Experimental results: example skill graph and topological order in ASSISTment dataset

two skills 'Ordering Fractions' and 'Ordering Positive Decimals', which actually have no clear prerequisite relationship between these two skills in reality.

Another interesting observation is that strongly associated skills described in the skill graph occurred far apart in the input exercise sequences. For example, '*Ordering Integers*' is a prerequisite of '*Ordering Fractions*'. However, even though these two skills do not appear in a sequence, '*Ordering Integers*' always appears to be a prerequisite of '*Ordering Fractions*' in our experiments.

5 Future Work

The method of topological order discovery at this stage is generated using some predefined rules. In the next step, we desire to create an end-to-end training model to learn topological order and integrate the topological information into the RNNs framework to improve the prediction accuracy of student performance.

Acknowledgement. The work described in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CUHK 14208815 of the General Research Fund), and 2015 Microsoft Research Asia Collaborative Research Program (Project No. FY16-RES-THEME-005).

References

1. Piech, C., Spencer, J., Huang, J., Ganguli, S., Sahami, M., Guibas,L., Sohl-Dickstein, J.: Deep knowledge tracing. In: NIPS (2015)
2. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
3. Corbett, A.T., Anderson, J.R.: Knowledge tracing: modeling the acquisition of procedural knowledge. User Model. User-Adap. Interact. **4**(4), 253–278 (1994)
4. Pardos, Z.A., Heffernan, N.T.: Modeling individualization in a Bayesian networks implementation of knowledge tracing. In: De Bra, P., Kobsa, A., Chin, D. (eds.) UMAP 2010. LNCS, vol. 6075, pp. 255–266. Springer, Heidelberg (2010)
5. Yudelson, M.V., Koedinger, K.R., Gordon, G.J.: Individualized Bayesian knowledge tracing models. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 171–180. Springer, Heidelberg (2013)
6. Pardos, Z.A., Heffernan, N.T.: KT-IDEM: introducing item difficulty to the knowledge tracing model. In: Konstan, J.A., Conejo, R., Marzo, J.L., Oliver, N. (eds.) UMAP 2011. LNCS, vol. 6787, pp. 243–254. Springer, Heidelberg (2011)
7. Graves, A.: Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850 (2013)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 17351780 (1997)
9. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85117 (2015)
10. Khajah, M., Lindsey, R.V., Mozer, M.C.: How deep is knowledge tracing? arXiv preprint arXiv:1604.02416 (2016)
11. Hecht-Nielsen, R.: Theory of the backpropagation neural network. In: International Joint Conference on Neural Networks, pp. 593–605. IEEE (1989)
12. Kahn, A.B.: Topological sorting of large networks. Commun. ACM **5**(11), 558–562 (1962)