qDKT: Question-Centric Deep Knowledge Tracing

Shashank Sonkar[1], Andrew E. Waters[1,2],
Andrew S. Lan[3], Phillip J. Grimaldi[1,2], Richard G. Baraniuk[1,2]
[1]Rice University, [2]OpenStax, [3]University of Massachusetts Amherst
ss164@rice.edu, aew2@rice.edu, andrewlan@cs.umass.edu, pjg3@rice.edu, richb@rice.edu

ABSTRACT

Knowledge tracing (KT) models, e.g., the deep knowledge tracing (DKT) model, track an individual learner's acquisition of skills over time by examining the learner's performance on questions related to those skills. A practical limitation in most existing KT models is that all questions nested under a particular skill are treated as equivalent observations of a learner's ability, which is an inaccurate assumption in real-world educational scenarios. To overcome this limitation we introduce *qDKT*, a variant of DKT that models every learner's success probability on individual questions over time. qDKT incorporates graph Laplacian regularization to smooth predictions under each skill, which is particularly useful when the number of questions in the dataset is big. qDKT also uses an initialization scheme inspired by the fastText algorithm, which has found great success in a variety of language modeling tasks. Our experiments on several real-world datasets show that qDKT achieves state-of-art performance predicting learner outcomes. Thus, qDKT can serve as a simple, yet tough-to-beat, baseline for new question-centric KT models.

1. INTRODUCTION

Knowledge tracing (KT) models are useful tools which provide educators with actionable insights into learners' progress [21, 16]. Given a learner's performance history, these methods predict their proficiency across a predetermined set of skills (i.e., knowledge components or concepts). One of the most popular methods for tracking this cognitive development is the Bayesian Knowledge Tracing (BKT) framework [3, 15, 24] which applies hidden Markov models [1] to learn each learner's *guess, slip,* and *learn* probabilities for each skill. Another approach to modeling the dynamics of skill acquisition is SPARFA-Trace [11] which uses Kalman filtering [9] to model learner skill acquisition. An advantage of SPARFA-Trace is that, unlike BKT models, it can relate individual questions to multiple skills. Recently, deep learning techniques have been applied to the KT problem to create Deep Knowledge Tracking (DKT) [18] which models the sequence prediction task using a Long Short-Term Memory (LSTM) network [8].

All of the aforementioned KT models track an individual learner's knowledge at the *skill* level. Under the KT framework, the time series data modeled consists of learner skill interaction sequences, given by $X_i = \{(s_t^i, a_t^i)\}_{t=1}^T$ where s_t^i is the skill index attempted by the i^{th} learner at discrete time step t, while $a_t^i \in \{0, 1\}$ is the assessment of the learner's response, with 0 indicating an incorrect response and 1 indicating a correct response.

The key assumption underpinning all of the above models is that all questions nested under a particular skill are equivalent. This assumption, however, is generally unrealistic in real-world educational datasets. First, a mapping of questions to skills is not always available and obtaining such a mapping requires the intervention of subject matter experts, which is both costly and time-consuming. Second, questions in real-world educational datasets are never homogeneous, but rather exhibit significant variations in difficulty and discrimination [5]. In other words, different questions convey differing levels of information about a particular learner's mastery of the underlying skill, and methods for modeling learner's acquisition of skills over time should take such information into account.

However, simply substituting questions for skills in a traditional KT model is insufficient to accomplish the goal of tracking an individual learner's knowledge at the question level. To illustrate this, we selected two commonly used educational datasets, ASSISTments2009 and ASSISTments2017.[1] We first ran the standard DKT model using the skill-level information provided with each dataset. We then re-ran the DKT model but used the question identifiers themselves, rather than the skills, for modeling performance. Concretely, the time series data modeled consisted of learners' question interaction sequences, given by $X_i = \{(q_t^i, a_t^i)\}_{t=1}^T$, where q_t^i denotes the question answered by learner i at time t. The AUC for both of these model variants are shown in Table 1. We note that for the ASSISTments 2017 dataset that this question-centric approach provides a moderate improvement in AUC but for the ASSISTments 2009 dataset the question-centric approach significantly hurt AUC.

To understand why this behavior occurs, we note that the

[1]https://sites.google.com/site/assistmentsdata/home

Shashank Sonkar, Andrew Lan, Andrew Waters, Phillip Grimaldi and Richard Baraniuk "qDKT: Question-centric Deep Knowledge Tracing" In: *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020),* Anna N. Rafferty, Jacob Whitehill, Violetta Cavalli-Sforza, and Cristobal Romero (eds.) 2020, pp. 677 - 681

Dataset	Number of questions	Avg. Obs. per question	DKT (skill)	DKT (question)
ASSISTments 2017	1,183	145.76	0.72	**0.74**
ASSISTments 2009	16,891	19.27	**0.74**	0.68

Table 1: AUC scores for DKT vs. its variant with questions as indices. Using questions indices leads to overfitting when the number of observations per question is small.

average number of observations per question for the ASSISTments 2009 dataset is significantly smaller than that for the ASSISTments 2017 dataset. This results in the question-centric modeling overfitting to the data, which adversely affects predictive accuracy. In contrast, the ASSISTments 2017 dataset has a larger number of observations per question, which helps the question-centric DKT model to avoid overfitting.

It is apparent that question-level modeling has the potential to significant improve predictive accuracy in KT models as compared to skill-level modeling. However, simply substituting questions for skills in a KT model is insufficient to realize the gain. Addressing this challenge is the focus of our work.

Our main contributions are summarized as follows:

1. We propose a novel algorithm for question-level knowledge tracing, which we dub *qDKT*, that achieves state-of-the-art performance compared to traditional KT methods on a number of real-world datasets.

2. Our method utilizes a novel graph Laplacian regularizer for incorporating question similarity information into qDKT. Question similarity can be calculated using the skill information or using textual similarity measures if the dataset contains the actual text for each question. Unlike other KT methods, our method does not assume that each question must be associated with exactly one skill.

3. We propose a novel initialization scheme for question-level KT models using fastText [2], an algorithm for natural language processing (NLP). This initialization scheme learns embeddings that summarize pointwise mutual information statistics [12], which is beneficial for bootstrapping sequence prediction models.

Incorporating question-information to improve skill-centric KT models have been tried in the past, for example, the model proposed by [22] concatenates the question embedding to the skill embedding, which is then used as the input to the model. As training progresses, the model learns both the question embedding, and the skill embedding. However, the focus of our proposed initialization scheme is to bootstrap question-centric KT models without using any skill information. As stated earlier, this is advantageous because firstly, tagging questions with skills can be expensive, and secondly, the design of current skill-centric KT models does not transfer well to question-centric KT models (as shown in Table 1).

Initialized with the fastText-inspired scheme, qDKT performs at par with the state-of-art skill-level DKT model on ASSISTments 2009 dataset, and improves it by 5% and 6% on the ASSISTments 2017 dataset and Statics 2011 dataset respectively. Coupling the fastText-inspired scheme with the Laplacian regularizer, qDKT gives gains of 2% in AUC score as compared to the skill-centric DKT model for ASSISTments 2009, while also capturing question-specific characteristics.

2. PROBLEM STATEMENT AND DKT OVERVIEW

Each learner's performance record contains the questions attempted, time at which each question was attempted, and the assessment of each response (either correct or incorrect). Also, assume that the skill associated with every question is known. Given performance records for several learners, one wishes to train a knowledge tracing model with the objective of predicting the success probabilities across the questions (or the skills) at time T for a new learner whose performance history has been recorded until time $T-1$.

2.1 DKT Model

DKT uses an LSTM to predict a learner's future performance using their previous assessment history. As discussed earlier, the input to the model is a time series which consists of learners' skill interaction sequences, given by $X_i = \{(s_t^i, a_t^i)\}_{t=1}^{T}$. Here we restrict our discussion to a single learner and will omit the superscript i throughout. The forward equations of the DKT model are given by

$$\boldsymbol{x}_t = W_{xv}\boldsymbol{v}_t, \tag{1}$$

$$\boldsymbol{h}_t = LSTM(\boldsymbol{x_t}), \tag{2}$$

$$\boldsymbol{y}_t = \sigma(W_{yh}\boldsymbol{h}_t + \boldsymbol{b}_y), \tag{3}$$

where σ is the sigmoid function. In words, the input at time step t is the skill interaction tuple (s_t, a_t) which is encoded by an arbitrary high-dimensional one-hot vector, $\boldsymbol{v}_t \in \{0,1\}^{2M}$, where M is the number of skills. Using an embedding matrix, $W_{xv} \in R^{K\times 2M}$, $\boldsymbol{v}_t$ is mapped to a low-dimensional vector, $\boldsymbol{x}_t \in R^K, K \ll M$ (1), which serves as the input to the LSTM cell. $\boldsymbol{x}_t$ is passed through each of the input, forget, and output gates and, in the end, the LSTM returns $\boldsymbol{h}_t$ – the estimate of the learner's current knowledge state. The final output of the model is $\boldsymbol{y}_t \in R^M$ which predicts the learner's success probabilities for all the M skills for the next time step $t+1$.

2.1.1 Loss in the DKT Model

The output $\boldsymbol{y}_t$ of the DKT model predicts the learner's proficiency over the skills for the next time step $t+1$. During training, the assessment (a_{t+1}) of the learner's response to the question indexed by q_{t+1} is known beforehand. The success probability for the skill associated with q_{t+1} is given by $y_t[s_{t+1}]$. Since DKT assumes that mastery in the skill is equivalent to mastery in any of the questions under it (i.e., all questions under a skill are equivalent), a trained DKT model should predict the success probability at the skill to be the same as the assessment. This rationale motivates the basis for calculating the loss, ℓ_t, at time t, given by

$$\ell_t = l(y_t[s_{t+1}], a_{t+1}), \tag{4}$$

where ℓ is the binary cross-entropy loss.

2.2 Proposed Model: qDKT

We now introduce our proposed method for KT modeling at the question-level, which we dub qDKT. Our method considers a modified problem statement where we estimate a learner's success probability for each question rather than for each skill. Let a learner's question interaction sequence $X = \{(q_t, a_t)\}_{t=1}^{T-1}$ until time step $T-1$ be given, where q_t denotes the question answered at time t and $a_t \in \{0, 1\}$ is the assessment of the response to question q_t. Our goal is to output $\boldsymbol{y}_t \in R^N$ which predicts the learner's success probabilities for all the N questions at the next time step $t+1$. qDKT utilizes the same architecture as DKT as specified in (1) - (3), but with $\boldsymbol{v}_t \in \{0, 1\}^{2N}$, $W_{xv} \in R^{K \times 2N}$, and $\boldsymbol{y} \in R^N$. The updated loss ℓ_t from (4) at time t is then given by

$$\ell_t = l(y_t[q_{t+1}], a_{t+1}). \tag{5}$$

We eill refer to this model as the *base qDKT model*, where the prefix q denotes question-level modeling.

3. REGULARIZATION FOR qDKT

As seen in Table 1, the base qDKT model performs poorly for datasets with both a large number of questions and a small number of observations per question. To overcome this, we propose a regularization method for qDKT to combat overfitting. It is reasonable to assume that success probabilities of multiple questions associated with the same skill should not be significantly different for a given learner. Based on this premise, we regularize the variance in success probabilities for questions that fall under the same skill

$$R(\boldsymbol{y}) = \sum_{i \in Q} \sum_{j \in Q} \mathbf{1}(i, j) \cdot (y_i - y_j)^2, \tag{6}$$

where vector $\boldsymbol{y} \in R^N$ contains success probabilities of all questions Q in the dataset, $i, j \in Q$ and $\mathbf{1}(i, j)$ is 1 if i, j fall under the same skill, otherwise it is 0.

We add this penalty to the loss and use λ to control the weight of the penalty. Thus, the updated loss function from (4) with the regularization penalty is

$$\ell = l + \lambda \cdot R(\boldsymbol{y}). \tag{7}$$

3.1 Interpretation of the regularizer

Graph theory provides a clean interpretation for the regularization penalty which is also helpful for speeding up its computation. We construct a graph G with number of nodes equal to the number of questions in the dataset. Two nodes are connected with an edge of weight 1 if the questions are associated with the same skill and with an edge weight of 0 otherwise.

The degree matrix D of a graph G is a diagonal matrix with

$$d_{ii} = \sum_{j \in C_i} w_{ij},$$

where w_{ij} is the similarity between node i and node j (edge weight), C is the set containing all the indices directly connected with i (immediate siblings). The adjacency matrix A of a graph G stores the edge weights w_{ij}. Given the degree matrix D and the adjacency matrix A of a graph G, the Laplacian matrix L is defined as

$$L = D - A.$$

Then for any vector $\boldsymbol{v}$ [7],

$$\boldsymbol{v}^T L \boldsymbol{v} = \sum_{i,j} w_{ij} \cdot (v_i - v_j)^2. \tag{8}$$

We can then use (8) to simplify the regularization penalty of (6)

$$R(\boldsymbol{y}) = \sum_{i \in Q} \sum_{j \in Q} \mathbf{1}(i, j) \cdot (y_i - y_j)^2 = \boldsymbol{y}^T L \boldsymbol{y}. \tag{9}$$

The simplification of the double summation term to a condensed vector-matrix multiplication term is useful to speed up its calculation, especially while training the qDKT model on GPUs.

Further, our approach to model similarity works even when questions are associated with multiple skills. This provides additional flexibility over previous KT models that restrict each question to be associated to exactly one skill. Such flexibility is important for real-world applications where questions commonly evaluate learners on multiple skills simultaneously. Moreover, this formulation can be helpful to incorporate even other measures of similarity like tf-idf similarity [13] using question text.

4. INITIALIZATION OF qDKT

DKT maps each skill interaction tuple to $\boldsymbol{x} \in R^d$ via the matrix W_{xv} (see (1)). In DKT, the entries of W_{xv} are initialized with draws from a standard normal distribution. While this approach is straightforward, random embeddings tend to perform extremely poorly in high dimensions where the optimization problem will have an extremely large number of saddle points [4]. To overcome this limitation, we propose a more effective method for initializing W_{xv} inspired by the fastText architecture.

4.1 Language Modeling and fastText

In NLP, language models are used to predict the most likely words that can follow a given sequence of words. Such models are often initialized with word embeddings from algorithms like word2vec [14], fastText and GloVe [17]. At a high level, these algorithms embed words into a high dimensional space such that words that have close semantic relationships will be embedded near one another, while words with low semantic similarity will be embedded further apart [6].

A novelty of fastText is that it considers individual characters in a word when computing the final embeddings. By doing this, fastText recognizes that the words "love", "loved", "lovely", and "lovable" are all related and embed them accordingly.

4.2 Embedding Educational Response Data

In our application, we wish to have a notion of question similarity that can serve to guide our initialization scheme, similar to the notion of similar word contexts in fastText.

Dataset	Learners	Questions	Skills	Records
ASSISTments 2009	4,151	16,891	111	325,637
ASSISTments 2017	1,709	1,183	86	249,105
Statics2011	333	1,223	85	189,297
Tutor	895	5981	1,592	437,524

Table 2: Dataset summary statistics.

To do this, we assemble an approximate "text corpus" from our response data, as follows.

Let set Q contain all the question ids and set U contain all characters. We define a one-to-one mapping $f : Q \to U$ which maps a question id to a unique character. To convert learners' question interaction sequences, $X = \{(q_t, a_t)\}_{t=1}^{T}$ into a text corpus, we apply a signal transformation Y on X such that $y_t = f(q_t) + a_t$ where '+' denotes the string concatenation operator. Thus, each question interaction is encoded as a two character string consisting of the question id and the graded response. This interaction encoding constitutes the "words" of our corpus. The "sentences" of our corpus constitute of the string of such encoded interactions by an individual learner. We finally apply fastText to this newly generated "corpus". For a given question interaction say $(q, 0)$, fastText will train the embeddings of the following n-grams $\{f(q), '0', f(q) + '0'\}$. Thus, we link the embeddings of $(q, 0)$ and $(q, 1)$ through the embedding of $f(q)$. The resulting output embedding of fastText is used as our initialization of W_{xv}.

5. EXPERIMENTS

5.1 Datasets

We consider four datasets for our experiments: ASSISTments 2009, ASSISTments 2017, Statics 2011, and a dataset from OpenStax Tutor, an online learning platform. The Statics 2011 dataset is from an engineering statics course. Standard pre-processing steps common in the literature are used to clean the data. For ASSISTments2009 dataset, we follow the pre-processing steps recommended by [23]. Duplicated records and scaffolding problems are removed. Also, since the dataset contains a few questions that are associated with multiple skills, those multiple skills were combined into a new joint skill for skill-level DKT models, along the lines of [23]. However, for qDKT, our Laplacian regularization approach provides needed flexibility when questions fall under multiple skills, doing away with the need of combining multiple skill into one joint skill. For the ASSISTments2017 dataset, all scaffolding problems are filtered out. Relevant statistics for each dataset are given in Table 2.

5.2 Experimental Setup and Metrics

Each experiment consists of comparing our proposed qDKT algorithm against the original DKT algorithm for a given dataset. To further quantify the impact of each proposed improvement to the qDKT model we will measure qDKT performance over four different variants: 1) The base qDKT without any regularization and with randomized initialization, 2) qDKT with regularization and randomized initialization, 3) qDKT without regularization but with our proposed initialization scheme and 4) qDKT with both regularization and with our proposed initialization scheme. For all the experiments and datasets, we perform 5-fold cross validation; 70% data is used for training and the rest for testing. We report the average receiver operating characteristics curve (AUC) score to compare each method. All the models are trained using the Adam optimizer [10] with dropout [20] to reduce overfitting.

5.3 Results and Discussion

Our results are displayed in Table 3. We see that the base qDKT model without regularization and with randomized initialization outperforms the original DKT model on three of the four datasets used. For the ASSISTments 2009 dataset, base qDKT loses by a large margin. This is due to ASSISTments 2009 dataset having a large number of questions coupled with a low number of observations per question (see Table 1). We note that the individual addition of either the regularizer or the fastText initialization scheme greatly improves the performance of qDKT for each dataset. We finally note that the combination of both the regularizer and fastText initialization scheme enables qDKT to achieve better performance than DKT for all datasets considered.

For additional details, please refer to the extended version of this paper [19].

6. CONCLUSIONS

We have proposed qDKT, a novel model for knowledge tracing for educational data. Our method improves on prior art by predicting student performance at the question-level, rather than at the skill level. We have further proposed novel regularization and initialization schemes that greatly improve the performance of our method across several real-world datasets when compared with the traditional knowledge tracing methods. We propose that qDKT can provide a simple, yet tough-to-beat baseline, for new question-centric KT models to come.

Acknowledgements

This work was supported by NSF grants CCF-1911094, IIS-1838177, IIS-1730574, DRL-1631556, IUSE-1842378, NSF-1937134; ONR grants N00014-18-12571 and N00014-17-1-2551; AFOSR grant FA9550-18-1-0478; and a Vannevar Bush Faculty Fellowship, ONR grant N00014-18-1-2047.

7. REFERENCES

[1] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

[2] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

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

[4] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems*, pages 2933–2941, 2014.

Dataset	DKT	Base qDKT	Base qDKT w/ Laplacian regularizer	Base qDKT w/ fastText	Base qDKT w/ fastText and regularizer
ASSISTments 2009	0.740 ± 0.002	0.678 ± 0.004	0.738 ± 0.003	0.740 ± 0.004	**0.762 ± 0.005**
ASSISTments 2017	0.721 ± 0.002	0.742 ± 0.003	0.753 ± 0.005	**0.772 ± 0.004**	0.770 ± 0.005
Statics 2011	0.770 ± 0.003	0.822 ± 0.003	0.825 ± 0.002	0.832 ± 0.003	**0.834 ± 0.002**
Tutor	0.856 ± 0.003	0.875 ± 0.002	0.882 ± 0.001	0.890 ± 0.0008	**0.895 ± 0.001**

Table 3: AUC scores for each algorithm and dataset. We see that both the addition of the regularizer and the improved initialization scheme improve performance on all datasets over the original DKT model. Combining both the regularizer and our proposed initialization scheme achieves the best performance over all algorithms.

[5] S. E. Embretson and S. P. Reise. *Item response theory*. Psychology Press, 2013.

[6] Y. Goldberg and O. Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[7] D. J. Hand. Statistical analysis of network data: Methods and models by eric d. kolaczyk. *International Statistical Review*, 78(1):135–135, 2010.

[8] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[9] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

[11] A. S. Lan, C. Studer, and R. G. Baraniuk. Time-varying learning and content analytics via sparse factor analysis. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 452–461. ACM, 2014.

[12] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185, 2014.

[13] J. H. Martin and D. Jurafsky. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Pearson/Prentice Hall Upper Saddle River, 2009.

[14] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[15] Z. Pardos and N. Heffernan. Navigating the parameter space of bayesian knowledge tracing models: Visualizations of the convergence of the expectation maximization algorithm. In *Educational Data Mining 2010*, 2010.

[16] R. Pelánek. Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques. *User Modeling and User-Adapted Interaction*, 27(3-5):313–350, 2017.

[17] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[18] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *Advances in neural information processing systems*, pages 505–513, 2015.

[19] S. Sonkar, A. E. Waters, A. S. Lan, P. J. Grimaldi, and R. G. Baraniuk. qdkt: Question-centric deep knowledge tracing. *arXiv preprint arXiv:2005.12442*, 2020.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[21] K. Vanlehn. The behavior of tutoring systems. *International journal of artificial intelligence in education*, 16(3):227–265, 2006.

[22] T. Wang, F. Ma, and J. Gao. Deep hierarchical knowledge tracing. In *EDM*, 2019.

[23] X. Xiong, S. Zhao, E. G. Van Inwegen, and J. E. Beck. Going deeper with deep knowledge tracing. *International Educational Data Mining Society*, 2016.

[24] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *International conference on artificial intelligence in education*, pages 171–180. Springer, 2013.