Deep Embeddings of Contextual Assessment Data for Improving Performance Prediction

Benjamin Clavié
School of Informatics, University of Edinburgh
Edinburgh, Scotland
benjamin.clavie@ed.acuk

Kobi Gal
School of Informatics, University of Edinburgh
Edinburgh, Scotland
kgal@ed.ac.uk

ABSTRACT

We introduce DeepPerfEmb, or DPE, a new deep-learning model that captures dense representations of students' online behaviour and meta-data about students and educational content. The model uses these representations to predict student performance. We evaluate DPE on standard datasets from the literature, showing superior performance to the state-of-the-art systems in predicting whether or not students will answer a given question correctly. In particular, DPE is unaffected by the cold-start problem which arises when new students come to the system with little to no data available. We also show strong performance of the model when removing students' histories altogether, relying in part on contextual information about the questions. This strong performance without any information about the learners' histories demonstrates the high potential of using deep embedded representations of contextual information in educational data mining.

1. INTRODUCTION

The *testing* effect, the effect of including practice assessments as part of a students' learning phase, is known to have a strong positive influence on the knowledge acquisition process [2].

While the importance of regular practice and question answering is established, it is essential to balance it against the time constraints that students and instructors are facing [11]. The issue of having to teach and evaluate *"too much [...] in too short a time"* [10] is long-standing and leads to teachers having to make instructional choices with the information they have available [12]. It is thus important to identify factors that could help intelligent systems to ask the right question to the right students to maximise their knowledge gain in a limited time.

Extensive research has focused on building better student modeling to work towards this goal. Most of these approaches focus on extracting information from individuals' histories of answers given, both right and wrong, to questions evaluating certain skills [4, 19, 18, 7]. Recent work has taken into account other factors, such as item-skills relationships, the relationship between a question and the skill it is meant to evaluate citedas3h, or individual item difficulty [17] in predicting student performance .

Deep knowledge tracing, which represents the state of the art in student performance, does not take into account the wealth of instance-specific interactions a student has with a given question, such as requesting assistance before attempting to answer it or the amount of time taken before answering.

We propose DeepPerfEmb, a deep learning model whose aim is to learn dense representations of this information and use it to improve the task of performance prediction. Our contribution is two-fold: we firstly argue that instance-specific information can be leveraged by such a model to reach a very high level of performance on predicting student correctness. We also introduce a variant of the model using exclusively contextual data, showing its ability to learn dense representations of these data points and perform strongly on the same task, despite having very limited information about the students' actions.

2. BACKGROUND

In the educational data mining field, there has been extensive research on attempting to model a learner's understanding of defined skills. Generally, this task is achieved through using observations related to a student's question-answering history. This information is used to estimate the student's mastery of the skills evaluated by the questions and is generally evaluated by using the model to predict whether or not they will answer a given question correctly. Such models are known as *Knowledge Tracing* (KT) models. *Bayesian Knowledge Tracing* (BKT), one of the most widespread classical method, models each students' knowledge as the latent variable of a Hidden Markov Model built using students' answering histories [4]. Such methods also rely on an evaluation of the probability of *slipping*, when a student answers incorrectly despite having mastered the skill, and *guessing*, when a correct answer is given without having mastered it.

More recently, many different approaches to knowledge tracing have been researched, mainly relying on extracting information from a vast amount of students' attempts at answering questions [7, 18]. Some of these models occasionally

Benjamin Clavié and Kobi Gal "Deep Embeddings of Contextual Assessment Data for Improving Performance Prediction" 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. 374 - 380

focus on or integrate other factors, such as modelling student forgetting [23] and estimated difficulty of question [15] or the possibility for a single question to relate to multiple, distinct types of knowledge [3]. These approaches often serve as the basis to intelligent learning schedulers, aiming to optimise the distribution of questions asked to students to maximise their knowledge gain [22, 25].

In recent years, deep learning has been utilised in order to produce better-performing variants of previous approaches. Notably, DeepIRT [27] and Deep Knowledge Tracing [19], have been introduced. These techniques, themselves a refinement on previous models, replace some of the prior building blocks with deep neural architectures while retaining the same foundational approach. Unlike more traditional methods, deep-learning based approaches rarely explicitly model the impact of forgetting, guessing or slipping, instead relying on the model to capture implicit information about these factors.

Online intelligent tutoring systems, such as the Assistment platform[20], have been invaluable in providing a large amount of data to train and evaluate such models. In addition to the information about students' attempts, failures and successes in answering questions, they generate a wealth of data about other aspects of the tutoring system. Notably, such systems may provide the user with the possibility of requesting assistance in answering the question, in the form of hints. It has been noted that such additional features are under-utilised in KT models and improve their performance when taken into account [26].

The focus of most of this prior work has been on exploiting the history of user answers, both right and wrong, in order to predict the likelihood that they have mastered a given skill. Such approaches reach a high level of performance and can accurately model the relationships between the skills evaluated [19, 16]. However, they encounter issues with students with relatively little or no interaction, and some of them exclude any student who has attempted to answer fewer than 10 questions [15, 3]. This issue is known as the *cold start problem*.

However, point-of-time snapshots of data contain a lot of additional information that has known little exploration. Such information, which we broadly refer to as **contextual information**, includes data directly related to the students' context, such as their school, the question they are solving, and the time it takes them to attempt to answer a question. We believe that such a method is complementary to approaches focusing on students' history in understanding the cognitive process of learning through assessment.

Prior work on deep neural networks has highlighted their ability to learn good embedding representations for discrete data [6]. This paper demonstrates that a modified version of this approach is able to outperform state-of-the-art KT model in the specific task of predicting student correctness. We show that our model learns a powerful representation of the data it receives as input, outperforming the state of the art, leading to a better understanding of how the questions asked to students can affect their performance.

3. PROPOSED METHOD

Our goal is to highlight how contextual data can be leveraged to improve question-correctness prediction. In order to do so, we use a deep learning model whose main purpose is to learn representations of this data in order to predict question-correctness. We then set out to leverage interpretation methods in order to understand which factors are considered important in making these predictions.

3.1 Data

We use two widely used public datasets made available by the Assistments online tutoring platform [20]: **ASSIST2009** [5] and **ASSISTChall** [1].
Each dataset is composed of hundreds of thousands of student interaction, with each interaction corresponding to a snapshot taken at the moment a student attempts to answer a question. Each snapshot contains a large amount of information, represented by multiple variables.

Two categories of data are present in each snapshot:

- **Meta-data**, or **contextual data**: Information about the overall context around the student and the question they are currently taking. Broadly, these are:
  - Information about the student's background (school ID, teacher ID...)
  - Information about the current question (problem set ID, question ID, skill evaluated ID, whether or not the question can be *scaffolded*...)
- Current **instance-specific data**: Information about the question the student is currently attempting. Broadly, these are:
  - Information about the student's help requests (hints requested, whether he has seen the final hint, where the questions stands in a *scaffolding*...)
  - Information about the time spent on the current question (time before first interaction, total time with question...)

Both datasets do not contain exactly the same information. ASSIST09 contains additional information in the form of both **interaction data**, such as time-to-first action and total time on question, and **contextual meta data**, notably relative to individual students' background, such as the specific assignment set they are working on or the ID of their class. Additionally, ASSISTChall is notable due to the presence of **scaffolded** questions. Scaffolding is an alternative to hints in making it simpler for a student to answer a harder question [21]. A scaffolded question is a question that can be decomposed into simpler questions (the *scaffolding questions*). The data contains variables describing the scaffolding status of an interaction: whether a question is the start of a scaffolding and whether it is part of one.
For the purpose of our experiments, we consider scaffolding to be a type of contextual data as an attribute of the question being asked.

Due to the nature of the information contained in our snapshots, they contain both **categorical** and **continuous** variables:

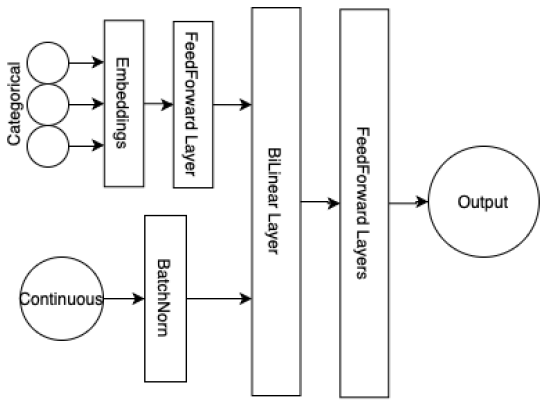

Figure 1: Simplified view of full model

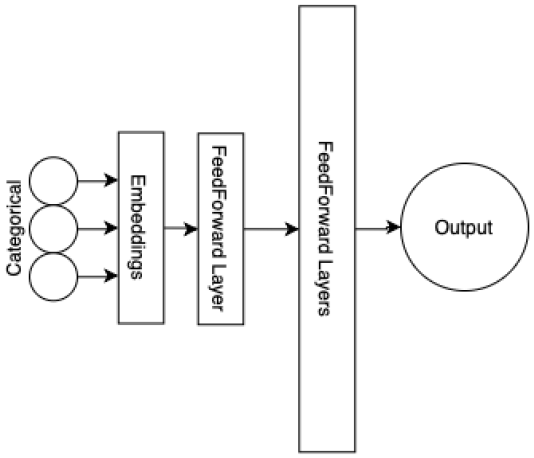

Figure 2: Simplified view of meta-data only model

Categorical variables, in this case, represent information that belongs to a finite number of defined categories, such as the skill being evaluated, the ID of the problem set the student is working through or the first action that they took on the current question (whether he requested a hint or attempted to answer it).
Continuous variables, on the other hand, represent information that can be measured, such as how long it takes for the user to first interact with the question after seeing it. For this work, ordinal variables, such as how many hints a student has received, are treated the same as continuous variables.

3.2 Preprocessing

We apply four major preprocessing steps to the data. For all of them except the removal of non-attempt snapshots, we use the data preprocessing utilities in the fastai2 library [8].

3.2.1 Removal of information leaks

Both datasets contain some variables that are perfectly correlated with student correctness. These are values such as the **hint** variable, which indicates that this interaction resulted in the user requesting a hint instead of trying to answer the question. The system will automatically label this interaction as "incorrect", although no attempt was made. As we do not want the model to learn incorrect information from this data and reach an artificially high score, these interactions are removed from the data.

Additionally, we also remove the variables that could lead to our model learning about an individual's student history. This includes the user ID, the total count of attempts by a user, the exact timestamp of interactions as well as additional information contained in ASSISTChall, such as a student's career path, final test score or emotional state.

3.2.2 Standardisation of Continuous Data

All the continuous variables are normalised before being fed to the model.

3.2.3 Handling Missing Continuous Value

In some cases, all continuous variables are not available in a given snapshot. In order to account for this factor, we create a categorical variable corresponding to each continuous variable. This variable represents whether the information is present in the current snapshot or not. This allows the model to potentially capture the meaning of the absence of a given observation in a snapshot.

3.2.4 Pre-encoding of Categorical Data

Prior to being passed as input to the model, all categorical variables are ordinally encoded. This means that each possible value is replaced by an integer representing it. This step is crucial in ensuring the model can learn a dense representation of each possible value during training.

3.3 Model

Predicting the performance of a student based on a student's previous answers on questions meant to evaluate defined skills has been widely explored in work on *Knowledge Tracing*. Our aim is to build a model learning good representations of data without individual students' histories to predict whether or not a student will answer a question correctly.

Our model is a variant of the model presented in [6] with several modifications. The overall architecture can be described as follows.

3.3.1 Architecture

Structure

Embeddings: We create an embedding layer for each of the categorical variables we are processing. This embedding process uses a function e_i, which maps each possible categorical input x_i to a corresponding dense vector X_i:

$$e_i : x_i \mapsto X_i \tag{1}$$

This means that each of the categorical variables C will be mapped to a vector space. Each embedding is learned during the model training, and our aim is for the model to learn a representation of the categorical variables describing a given snapshot.

This step is the key step of our network, as the embeddings are trained alongside the full network during model training. With the task of predicting student correctness as its final objective, the model will use these embeddings layer to learn a representation for each of the variables it is given as an input.
Finally, the embedded representation of all the categorical variables are concatenated together into a single vector. This vector is then passed to a single feedforward layer, as defined below.

Bilinear Layer: The authors of [6] concatenated the normalised continuous inputs with the previously generated concatenation of the categorical variables. This approach resulted in unstable training and overfitting on ASSIST09. To alleviate this and allow our model to better weigh both types of features, we introduce a Bilinear layer.
The Bilinear layer takes two vectors as input, x and y, and turns them into a single output vector by multiplicating them with a learned weight w and adding a learned bias b. The activation function and batch normalisation functions are both applied to this and every subsequent layers:

$$BatchNorm(Mish(x * w * y + b)) \quad (2)$$

FeedForward layers: The inputs are then passed through a classical feedforward architecture made up of linear layers which multiply the single input vector x by a learned weight w and add a learned bias b:

$$BatchNorm(Mish(x * w + b)) \quad (3)$$

Output layer: Our output layer is a normal feedforward layer with two output nodes, representing the prediction made by the model (correct or incorrect).

For the experiments exploiting both interaction and meta-data, we use the full version of our model as presented in Figure 1. When using only the meta-data, which is expressed through categorical variables exclusively, we do not need the weighing introduced by the Bilinear layer to allow the model to converge. As a result, in this situation, we use a simplified architecture presented in Figure 2.

Information
Activation: Our model uses the Mish activation function, which has been shown to consistently outperform common activation functions such as ReLU [14].
Batch Normalisation: It has previously been demonstrated that batch normalisation helps in both stabilising and speeding up the training of neural networks [9]. As such, we apply batch normalisation to our continuous input and to the output of every other layer.
Dropout: To prevent overfitting, which happens when the model learns too much about the training data and fails to generalise, dropout [24] is applied after every layer. We applied a dropout value of 0.4 during our experiments.

4. EXPERIMENTAL SETTING

We separate our experimentation into two parts. Firstly, we will use both of the data types we defined earlier, **meta-data** and **instance specific data**. This experiment will serve as a first indicator of our model's ability to extract information from the data and build efficient representation. We will then perform feature importance analysis on the models' predictions to understand what variables have the strongest impact on its predictions.

Following this, we will attempt to predict question-correctness using exclusively **meta-data**. The aim of this experiment is to highlight how much the model can learn while using no information about the current assessment session or the learner's history. We will then study the model to understand what representation of the data it has learned and how it impacts its performance.

We evaluate our model by performing 5-fold cross-validation and training the model for 100 epochs on each of the steps, saving and reporting the result obtained for the best epoch. For both datasets, we use the LAMB optimiser [29], which is better suited to large-batches training than other optimisers. In order to minimise training time, batch size is set to 24 000 and a maximum learning rate of 10^{-1} is used. In both models, we set the hidden dimensions of all layers to 100. These hyperparameters were obtained by a search using the first fold of the cross-validation set.

Due to the imbalanced distribution of our data, we report prediction results using the Area Under the receiver-operator Curve (**AUC**) metric, widely used in the literature for similar tasks [19, 3, 28].
For reference purposes, we have included results from the two most widespread implementations of Knowledge Tracing, BKT and DKT (here, DKT+ [28], a slight refinement of standard DKT) as well as from the current state-of-the-art, SAKT [16] in the comparison tables. For BKT, we use the best results reported in the paper introducing DKT [19].
Although the original data used by DPE and KT models is the same, we use different information found in the datasets. KT models use individual students' interaction histories in order to predict performance and discard the rest of the information. On the other hand, DPE focuses on the contextual data and explicitly avoids the use of any student history data. As such, the scores are given in order to compare their results when focusing exclusively on the task of predicting question-correctness, but are not directly comparable as KT models leverage this task as a way to model student behaviour whereas our aim is to evaluate the importance of other, individual-unrelated features.

4.1 Using Instance Specific and Meta-Data

We first attempt to build a performance predictor using the two types of data we defined earlier, **contextual meta-data** and **instance specific data**. This model is likely to perform well, as it has access to a vast array of information about the current question as well as instance information such as the amount and type of help requested, the time before an action is taken as well as the total time spent on the current question.

4.2 Using Meta-Data

Our second experiment focuses on using exclusively the data we defined earlier as **meta-data**. This means that we remove interaction data from the input data.

Table 1: Results

	All-data		Meta-data	
Model	ASSIST2009 AUC	ASSISTChall AUC	ASSIST2009 AUC	ASSISTChall AUC
DPE (Ours)	**0.87**	**0.76**	0.75	0.63
BKT (reference)	0.69	N/A	0.69	N/A
DKT+ (reference)	0.82	0.73	0.82	**0.73**
SAKT (reference)	0.84	0.73	**0.84**	**0.73**

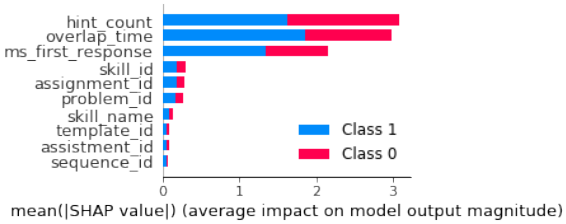

Figure 3: SHAP Values for ASSIST09 (all data)

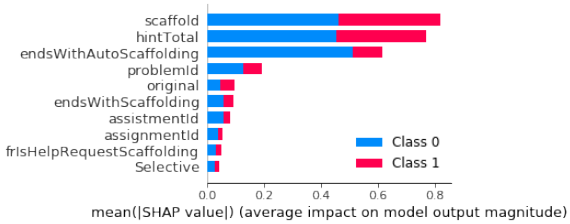

Figure 4: SHAP Values for ASSISTChall (all data)

We do so in order to force the classifier to learn strong representations of contextual meta-data about the student and the question themselves. Reaching a good level of performance using such limited data would suggest that these representations could be exploited to discover new insights about assessment and be combined with traditional knowledge tracing techniques to develop better assignments.

4.3 Interpreting Results And Feature Importances

Following the evaluation of the classifiers, we will attempt to extract information about the factors that strongly influence our model.
We will interpret the model's predictions using Deep Shapley Additive Explanations (DeepSHAP) [13]. By randomly replacing the values of subsets of the input features by uninformative values, DeepSHAP measures the influence of each input feature on different parts of a deep neural network and produces SHAP values for each prediction examples. SHAP values are an estimation of the importance of the feature in the prediction of each label made by the model.
We run DeepSHAP on randomly selected representative examples from the validation set and report the mean SHAP values of the features over all the examples, providing a visualisation of the features used by the model in its prediction. In all figures, class 0, the negative class, refers to a student answering a question incorrectly while class 1 refers to them having successfully answered the question. Although deep learning models remain black boxes and such interpretation techniques are vulnerable to adversarial examples, they provide a solid base towards making sense of model predictions.

5. RESULTS AND DISCUSSION

The results for this experiment are presented in Table 1, with BKT, DKT+ and SAKT results also presented for reference purposes.

When using all the available data, our approach performs extremely well in predicting question-correctness on ASSIST2009, reaching an AUC of 0.87 on ASSIST2009 and 0.76 on ASSISTChall, slightly outperforming state-of-the-art KT approaches for this task.
Our approach also reaches relatively high AUCs scores of 0.75 and 0.63 on ASSIST09 and ASSISTChall, respectively, when removing the instance-specific interaction data and using meta-data exclusively. This suggests that the models, while not outperforming student history-based methods, are able to extract enough information from contextual meta-data to reach a good level of performance, even outperforming the reported BKT results for ASSIST2009.
In order to better understand what factors drive the models' performance, we will compute the SHAP value corresponding to an estimate of the importance of each feature.

The SHAP values for the models exploiting the full data are presented in Figure 3 and 4. In ASSIST09, the temporal features, detailing how long the student has been interacting with the current question and how long until they first interact with the question, are of high importance.
More notably, on both datasets, the features that appear to be the most influential focus measuring the amount of help a student has needed to answer the current question. Features related to **hints**, such as the amount of hints requested for the current question (*hint_count* and *hint_total*), have a very strong influence on predictions. As hints are automatically given in case of failure, the hint-related features also capture information about the number of attempts made on the current question during the current question.
In ASSISTChall,features related to **scaffolding**, another form of assistance the student can receive, also have strong influence on the prediction, further supporting the importance of assistment factors.
The figure also shows that the other variables which we described as **meta-data**, such as the problem ID, do play a role predicting question-correctness, with a stronger impact on the likelihood of a question being answered incorrectly than correctly. We explore the influence of these factors further in Figure 5 and 6, showing SHAP values for the models which only use contextual meta-data.

In the case of ASSISTment, we notice that problems with the ability to end in auto-scaffolding are a strong predictor on whether or not a student will correctly answer a question.

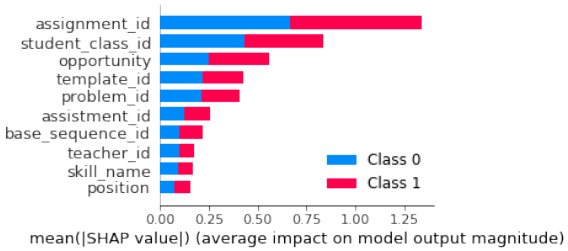

Figure 5: SHAP values for ASSIST09 (meta-data)

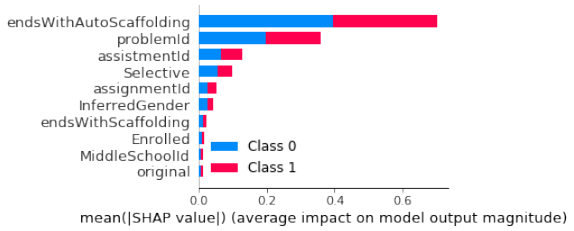

Figure 6: SHAP values for ASSISTChall (meta-data)

This is on par with our previous results, having shown the importance of assistance in predicting correctness. A possible explanation to this high impact on prediction is that questions with built-in scaffolding are likely to be of a higher difficulty level, leading the instructor to include scaffolding questions. Likewise, *original*, indicating a question isn't part of a scaffolding, has a moderately strong impact.
Besides scaffolding, both models rely on contextual information about the questions, such as the ID of the problem set or the ID of the problem itself. In ASSIST09, the additional information about the students' background, represented by their class and teacher IDs, is shown to be important to the predictive ability of the model.

The strong results achieved by these models, with very little information about the user's studies and history of previous answers, highlight the value of the representations the model learned. Without relying on user-success history, this contextual meta-data only model is able to reach a high AUC score, even outperforming the classical BKT approach on ASSIST09. This further reinforces the potential of integrating novel techniques to leverage contextual information when evaluating student mastery rather than relying solely on students' answers history.

6. CONCLUSION AND FUTURE WORK

In this paper, we introduced a novel deep learning model able to efficiently learn deep representations of contextual assessment information.
We showed that the proposed model reaches a very high level of performance when using both meta and instance-specific data on predicting whether a student will correctly answer a question or not.
We further showed that we can reach a relatively high level of performance on the same task while using exclusively contextual meta-data and very limited student-related information.
Additionally, our analysis of the information learned by the model shows that there is valuable insight to be extracted from analysing its predictions.
This work highlights the potential of learning from contextual data on top of user-history data and could be extended in several ways.
Future work should focus on integrating such learned representations within traditional knowledge tracing systems and learning schedulers and comparing their predictions to those of DPE. Contextual information is complementary to the information these systems exploit and could lead to improvements in the learning process. We also intend to investigate how the results we have obtained could be used to enrich theory-grounded models such as DeepIRT [27].
Furthermore, such an approach opens the way to extending current systems with additional external information, such as information about a user's interaction with course materials surrounding the knowledge evaluated.

7. REFERENCES

[1] The 2017 assistments datamining competition. https://sites.google.com/view/assistmentsdatamining.
[2] O. O. Adesope, D. A. Trevisan, and N. Sundararajan. Rethinking the use of tests: A meta-analysis of practice testing. *Review of Educational Research*, 87(3):659–701, 2017.
[3] B. Choffin, F. Popineau, Y. Bourda, and J. Vie. DAS3H: modeling student learning and forgetting for optimally scheduling distributed practice of skills. In *Proceedings of the 12th International Conference on Educational Data Mining, EDM 2019, Montréal, Canada, July 2-5, 2019*, 2019.
[4] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4:253–278, 1994.
[5] M. Feng, N. Heffernan, and K. Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User Model. User-Adapt. Interact.*, 19:243–266, 08 2009.
[6] C. Guo and F. Berkhahn. Entity embeddings of categorical variables. *arXiv preprint arXiv:1604.06737*, 2016.
[7] R. K. Hambleton, H. Swaminathan, and H. J. Rogers. *Fundamentals of item response theory*. Sage, 1991.
[8] J. Howard and S. Gugger. Fastai: A layered api for deep learning. *Information*, 11(2):108, Feb 2020.
[9] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.
[10] B. Kaur and S.-F. Yap. Kassel project report–third phase. *Singapore: National Institute of Education.*, 1998.
[11] J. M. Keiser and D. V. Lambdin. The clock is ticking: Time constraint issues in mathematics teaching reform. *The Journal of Educational Research*, 90(1):23–31, 1996.
[12] Y. Leong and H. Chick. Time pressure and instructional choices when teaching mathematics.

Mathematics Education Research Journal, 23:347–362, 09 2011.

[13] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774, 2017.

[14] D. Misra. Mish: A self regularized non-monotonic neural activation function, 2019.

[15] M. C. Mozer and R. V. Lindsey. Predicting and improving memory retention: Psychological theory matters in the big data era. In *Big data in cognitive science*, pages 43–73. Psychology Press, 2016.

[16] S. Pandey and G. Karypis. A self attentive model for knowledge tracing. In *Proceedings of the 12th International Conference on Educational Data Mining, EDM 2019, Montréal, Canada, July 2-5, 2019*, 2019.

[17] Z. A. Pardos and N. T. Heffernan. KT-IDEM: introducing item difficulty to the knowledge tracing model. In J. A. Konstan, R. Conejo, J. Marzo, and N. Oliver, editors, *User Modeling, Adaption and Personalization - 19th International Conference, UMAP 2011, Girona, Spain, July 11-15, 2011. Proceedings*, volume 6787 of *Lecture Notes in Computer Science*, pages 243–254. Springer, 2011.

[18] P. I. Pavlik Jr, H. Cen, and K. R. Koedinger. Performance factors analysis–a new alternative to knowledge tracing. *Online Submission*, 2009.

[19] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 505–513. Curran Associates, Inc., 2015.

[20] L. Razzaq, M. Feng, G. Nuzzo-Jones, N. Heffernan, K. Koedinger, B. Junker, S. Ritter, A. Knight, C. Aniszczyk, S. Choksey, et al. The assistment project: Blending assessment and assisting. In *Proceedings of the 12th annual conference on artificial intelligence in education*, pages 555–562, 2005.

[21] L. Razzaq and N. T. Heffernan. Scaffolding vs. hints in the assistment system. In *International Conference on Intelligent Tutoring Systems*, pages 635–644. Springer, 2006.

[22] S. Reddy, S. Levine, and A. Dragan. Accelerating human learning with deep reinforcement learning. In *NIPS workshop: teaching machines, robots, and humans*, 2017.

[23] B. Settles and B. Meeder. A trainable spaced repetition model for language learning. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 1848–1858, 2016.

[24] 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.

[25] B. Tabibian, U. Upadhyay, A. De, A. Zarezade, B. Schölkopf, and M. Gomez-Rodriguez. Enhancing human learning via spaced repetition optimization. *Proceedings of the National Academy of Sciences*, 116(10):3988–3993, 2019.

[26] Y. Wang and N. T. Heffernan. The "assistance" model: Leveraging how many hints and attempts a student needs. In *Twenty-Fourth International FLAIRS Conference*, 2011.

[27] C. Yeung. Deep-irt: Make deep learning based knowledge tracing explainable using item response theory. In *Proceedings of the 12th International Conference on Educational Data Mining, EDM 2019, Montréal, Canada, July 2-5, 2019*, 2019.

[28] C.-K. Yeung and D.-Y. Yeung. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, pages 1–10, 2018.

[29] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. In *International Conference on Learning Representations*, 2019.