Towards Temporality-Sensitive Recurrent Neural Networks through Enriched Traces

Thomas Sergent
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
Lalilo, Paris, France
thomas.sergent@lip6.fr

François Bouchet
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
francois.bouchet@lip6.fr

Thibault Carron
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
thibault.carron@lip6.fr

ABSTRACT

Educational traces are distinctive compared to the usual data a recurrent neural network encounters: there is a difference between two consecutive educational traces generated by a same learner if they are separated by 2 minutes or 2 months. Indeed, in the latter case, the learner who generated the trace may have forgotten the associated skill, which is less likely in the former case. Recurrent Neural Networks have seen a surge of popularity in the recent few years thanks to Deep Knowledge Tracing. While the focus has mostly been on the network architecture, we propose here a novel framework where traces are enriched with information relative to the temporality before they are used to train the network, and assess the performance on two datasets (Lalilo and ASSISTments 2012), which is not improved by this approach.

Keywords

Educational Data Mining, Neural Networks, Trace enrichment, Temporality

1. INTRODUCTION

1.1 Modelling student learning

As reminded by Choffin et al. [3], there are two main approaches to model students' learning: knowledge tracing and factor analysis.

On the one hand, knowledge tracing approaches model students' learning over time by nature by taking into account the sequential order of traces. Historically, these approaches started with Hidden Markov Models (HMMs) which were particularly used for Bayesian Knowledge Tracing (BKT) [4]. More recently, Deep Knowledge Tracing (DKT) appeared and spread partly thanks to increases in computing power [13]. The key idea is to model skills mastered by the students using Recurrent Neural Networks (RNNs).

On the other hand, Factor Analysis models have been developed since the 1950s (cf. [15] for a recent synthesis). They rely on the idea of making explicit the factors that can have an impact on students' success on a given exercise. Training the model then consists in computing the weight of those various factors in success.

1.2 How Recurrent Neural Networks are usually used

Thanks to massive increase in GPUs computing power, research in artificial neural networks and deep learning has developed at a fast pace over the past decade [10]. In the EDM community, its first use came with DKT in 2014, which uses RNNs in order to continuously model students' learning over time. Since then, several variations of RNNs have been created to better model students learning, such as Dynamic Key-Value Memory Networks for Knowledge Tracing (DKVMN) [17] and more recently Deep Hierarchical Knowledge Tracing (DHKT) [16] and Knowledge Query Network [11]. However all those alternative models only work on trying to adapt the structure itself of the RNN.

Outside of the EDM community, today some of the main uses of RNNs include natural language translation, speech recognition and time series forecasting. Those three uses have one common point: the distance between two successive data is always the same (a single space between two words of a text, 25ms between two audio samples in speech recognition or a same time difference between two points in time series). Nonetheless, this property is usually not true on problems datasets generated by students using various learning platforms or intelligent tutoring systems (ITSs) which are commonly considered in the EDM community. Indeed, in this context two consecutive log entries could be separated by two minutes (for two exercises done during the same learning session) or two months (if the student stops using the learning platform for a while). And when this time is long, it is likely that the student has either significantly progressed on that skill (through work outside of the system, e.g. class work) or on the contrary they may have forgotten some previously mastered skills.

1.3 Beyond sequentiality: temporality

But there are reasons to think that taking into account time data can significantly improve success prediction for an exercise. The importance of modeling temporal aspects in analyses of learning has been well-established in the EDM and LAK communities [9]. In sequence mining approaches for instance, authors tend to take it into account by allowing gaps between actions [8] and/or with feature engineering

Thomas Sergent, François Bouchet and Thibault Carron "Towards Temporality-Sensitive Recurrent Neural Networks through Enriched Traces" 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. 658 - 661

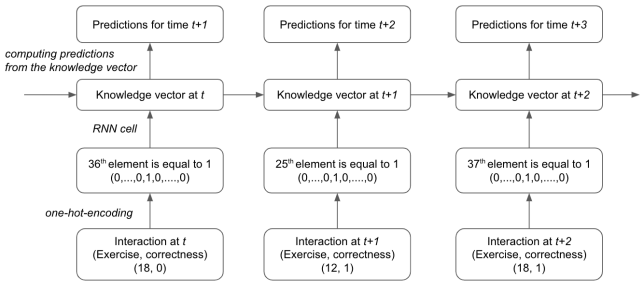

Figure 1: DKT structure. Adapted from [13]

by integrating this aspect in the actions themselves (short vs. long actions) [2]. The DAS3H model [3], which does not use RNNs, uses time windows of various durations to try to characterize the slopes of the forgetting curves for each skill. In deep learning approaches, Nagatani et al. [12] slightly modify the RNN structure to use exercise counts and time gaps between two similar exercises as additional inputs and are able to increase the AUC (Area Under ROC Curve) of predictions. In the medical field, a similar study to predict patient follow-ups benefited from a modification in the LSTM cell making it time-aware [1]. So likewise here we want to investigate ways to use the traces temporality and whether this could improve the quality of the predictions.

2. METHOD

2.1 Deep Knowledge Tracing in a nutshell

As mentioned before, the principle of DKT and its variants relies on an RNN whose weights characterize how a students' skills mastery evolves after an interaction with a learning system, such as an ITS. An interaction with the system is represented as a couple {exercise, answer}. So if there are K exercises with a boolean answer, $2K$ interactions are possible. In order to facilitate calculations in RNNs, they are usually one-hot-encoded using a binary vector of $2K$ values. For a sequence of N values, each interaction gets sequentially through the RNN. At a time t, the t-th interaction goes through the cell of the RNN thus providing the new vector representing the estimated knowledge of the student at that time. This vector can then be used to predict success on a given exercise at time $t+1$ (cf. Figure 1). Training a RNN thus corresponds to learning the transitions between a given student's knowledge vectors. We can notice that nowhere the temporal distance (or time gap) between two inputs is considered (cf. Figure 2 top), and as far as the authors know, no knowledge tracing algorithms are currently considering it.

2.2 Our proposal

Usually, enriching the traces consists in feature engineering and tends to be presented as an alternative to Recurrent Neural Networks [6]. [12] and [1] added temporality by modifying the structure of the neural network to include temporal information. Here we try a different approach by inserting new traces in the dataset, doing meta feature-engineering to be used by a Recurring Neural Network. Our idea consists in considering all traces from students as a sequence with missing values when there are no new trace for a given period of time. In cases like this, one can usually infer the missing values [7] by (1) adding data whose values are equal to the average of previous data or (2) adding again the same data that was last added. However, neither of these two approaches can be applied here. First, they are typically used for time series where only some variables are missing at time t but not all of them, whereas here it is equivalent to having all variables missing at time t. Moreover, averaging previous interactions does not make sense mathematically speaking. Finally, adding again the same interaction that was last added would not take into account the fact the student may have been progressing or forgetting during the time in which they were not using the learning platform.

	% of traces spaced by	
Dataset	> 7 days	> 30 days
ASSISTments12	3.1	0.6
ASSISTments17	1.1	0.4
Algebra I 2005-2006	0.3	0.02
Bridge to Algebra 2006-2007	0.02	0.003
Lalilo	2.0	0.4

Table 1: Dataset traces spread

Our proposal thus consists in adding traces (further on referred to as "artificial traces") at a regular predefined static time interval when students are not using the learning platform. The underlying hypothesis is that the RNN will be able to interpret those as time passing by. Those artificial traces are added as exercise $K+1$ (knowing there are only K exercises initially). Thus if we add traces every month without any exercise done, after 3 months without use, 2 artificial traces will have been added (cf. Figure 2 middle). In a similar scenario, if traces are added every week without any exercise done, 11 artificial traces will have been added (cf. Figure 2 bottom).

Adding those traces results in modifying the student's knowledge vector after each exercise $K+1$. After a given time without any new exercise done, the predictions corresponding to the probability of success will therefore differ from the ones without any artificial trace. When the RNN learns the meaning of that $K+1$-th exercise, it could lead to an improvement of the predictions.

In order to keep the initial tuple structure, we also add an arbitrary correctness of 0, which is not used practically to train the network.

3. EXPERIMENTS

3.1 Experimental setting

We computed the number of traces spaced by more than 7 and 30 days in a number of classical datasets : ASSISTments12 [5], ASSISTments17, Algebra I 2005-2006 and Bridge to Algebra 2006-2007. The two latter datasets stem from the KDD Cup 2010 EDM Challenge [14]. We were also able to get a dataset from Lalilo which is a web-app fostering literacy for K-2 (Table 1). In order to evaluate the performance of the traces enrichment, we have trained DKT and DHKT algorithms on the two datasets that had the highest spread in traces: ASSISTments12 (assist12) and Lalilo. Their main characteristics are summarized in Table 2.

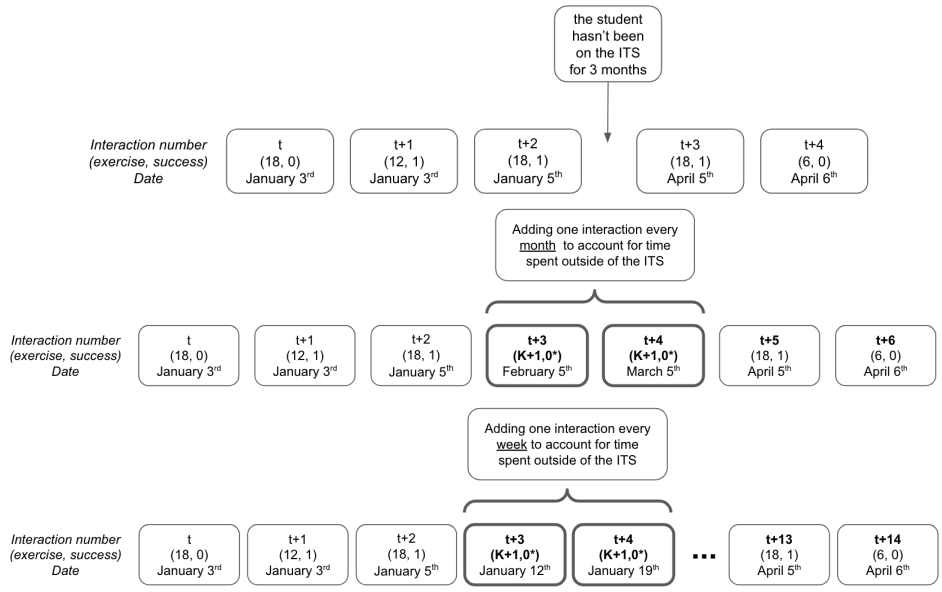

Figure 2: Traces fed to the network before enrichment (top), after *monthly* temporal enrichment (middle), or after *weekly* temporal enrichment (bottom)

Dataset	Users	Items	Skills	Interactions	Median length
lalilo	58,585	3,439	16	4,418,190	46
assist12	29,018	53,086	265	2,711,602	49

Table 2: Datasets characteristics

A key question with our approach relates to the frequency of use to add artificial traces. Indeed if the frequency is too high, it is likely that the artificial traces would disturb the RNN learning. Conversely, if the frequency is too low, it won't capture precisely the elapsed time. Therefore we compared the impact of various frequencies of artificial traces addition.

Our models have been implemented in Python and PyTorch for the Deep Learning aspects and the corresponding code is available online on GitHub[1]. Following Choffin et al. [3], we removed users for whom the number of interactions was less than 10 and interactions with NaN skills. We randomly sample randomly training (80%) and testing (20%) sets and give results on the testing set. We average on five different seeds and give standard deviation.

3.2 Results and analysis

A synthesis of the results can be found in the Tables 3 and 4. We use AUC to evaluate the performance of the models. No significant improvement in the predictions appear, and even with a high frequency of added artificial traces (daily), there is no significant degradation either. Several hypothesis could explain this lack of impact. It is possible that in those datasets, students are not progressing or regressing significantly between two moments when they use ASSISTments. A lack of differences would also be likely to be observed if long gaps without usage are unlikely. For example, if most students do 50 exercises over a few days, then stop using the system for 2 months, and use it again intensively for a week, the only exercises impacted would be the ones right after the gap of two months, i.e. only a small percentage of exercises overall. It is thus also possible that other datasets would be more sensitive to measure the impact of this artificial traces addition.

[1]https://github.com/thosgt/kt-algos

Model	Added trace frequency (# days)	AUC (std dev)
DKT	None	0.734 (0.004)
DKT	1	0.735 (0.003)
DKT	7	0.734 (0.002)
DKT	14	0.734 (0.002)
DKT	30	0.734 (0.005)
DHKT	None	0.771 (0.002)
DHKT	1	0.770 (0.002)
DHKT	7	0.771 (0.005)
DHKT	14	0.771 (0.003)
DHKT	30	0.770 (0.002)

Table 3: Performance comparison on the ASSISTments12 dataset

4. CONCLUSION AND PERSPECTIVES

We proposed here a framework to enrich learning traces to train recurrent neural networks. This enrichment which consists in adding artificial traces allows to add a temporality aspect into traces which normally only take into account se-

Model	Added trace frequency (# days)	AUC (std dev)
DKT	None	0.685 (0.001)
DKT	1	0.685 (0.003)
DKT	7	0.685 (0.002)
DKT	14	0.684 (0.003)
DKT	30	0.685 (0.002)
DHKT	None	0.701 (0.002)
DHKT	1	0.700 (0.002)
DHKT	7	0.701 (0.001)
DHKT	14	0.702 (0.003)
DHKT	30	0.700 (0.001)

Table 4: Performance comparison on the Lalilo dataset

quentiality. Unfortunately, ASSISTments 2012 and Lalilo datasets did not allow us to reveal a significant impact of our approach, but we have reasons to believe these particular datasets were not the most appropriate to measure a significant difference in the performance of prediction. Our future works thus involve (1) focusing on a population of students who has a scarce use of a learning platform over a large period of time (several months or years), (2) focusing on the impact of this algorithm over prediction specifically on exercises done right after a large time gap (during which the student may have learned or forgotten things), and (3) identifying learning platforms that teaches skills that are maybe easier to forget over time (e.g. vocabulary in a foreign language), or finding already existing datasets coming from such a platform.

5. ACKNOWLEDGMENTS

We would like to thank the reviewers for their thoughtful comments and efforts towards improving our paper. This work is funded by Lalilo.

6. REFERENCES

[1] I. M. Baytas, C. Xiao, X. Zhang, F. Wang, A. K. Jain, and J. Zhou. Patient Subtyping via Time-Aware LSTM Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 65–74, Halifax, NS, Canada, Aug. 2017. Association for Computing Machinery.

[2] F. Bouchet, R. Azevedo, J. S. Kinnebrew, and G. Biswas. Identifying Students' Characteristic Learning Behaviors in an Intelligent Tutoring System Fostering Self-Regulated Learning. *International Educational Data Mining Society*, 2012.

[3] B. Choffin, F. Popineau, Y. Bourda, and J.-J. Vie. DAS3H: Modeling Student Learning and Forgetting for Optimally Scheduling Distributed Practice of Skills. *EDM*, 2019.

[4] 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, Dec. 1994.

[5] M. Feng, N. Heffernan, and K. Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3):243–266, Aug. 2009.

[6] Y. Jiang, N. Bosch, R. S. Baker, L. Paquette, J. Ocumpaugh, J. M. A. L. Andres, A. L. Moore, and G. Biswas. Expert Feature-Engineering vs. Deep Neural Networks: Which Is Better for Sensor-Free Affect Detection? In *Artificial Intelligence in Education*, volume 10947, pages 198–211. Springer International Publishing, Cham, 2018.

[7] Y. J. Kim and M. Chi. Temporal Belief Memory: Imputing Missing Data during RNN Training. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 2326–2332, Stockholm, Sweden, July 2018. International Joint Conferences on Artificial Intelligence Organization.

[8] J. S. Kinnebrew and G. Biswas. *Identifying Learning Behaviors by Contextualizing Differential Sequence Mining with Action Features and Performance Evolution*. International Educational Data Mining Society, June 2012.

[9] S. Knight, A. F. Wise, B. Chen, and B. H. Cheng. It's about time: 4th international workshop on temporal analyses of learning data. In *Proceedings of the 5th International Conference on Learning Analytics and Knowledge, LAK 2015*, pages 388–389. Association for Computing Machinery, Mar. 2015.

[10] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[11] J. Lee and D.-Y. Yeung. Knowledge Query Network for Knowledge Tracing: How Knowledge Interacts with Skills. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, LAK19, pages 491–500, Tempe, AZ, USA, Mar. 2019. Association for Computing Machinery.

[12] K. Nagatani, Q. Zhang, M. Sato, Y.-Y. Chen, F. Chen, and T. Ohkuma. Augmenting Knowledge Tracing by Considering Forgetting Behavior. In *The World Wide Web Conference*, WWW '19, pages 3101–3107, San Francisco, CA, USA, May 2019. Association for Computing Machinery.

[13] 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. 00208.

[14] J. Stamper, A. Niculescu-Mizil, S. Ritter, G. Gordon, and K. Koedinger. Algebra I 2005-2006 and Bridge to Algebra 2006-2007. Development data sets from KDD Cup 2010 Educational Data Mining Challenge. Find them at http://pslcdatashop.web.cmu.edu/KDDCup/.

[15] J.-J. Vie and H. Kashima. Knowledge Tracing Machines: Factorization Machines for Knowledge Tracing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):750–757, July 2019.

[16] T. Wang, F. Ma, and J. Gao. Deep Hierarchical Knowledge Tracing. In *EDM*, 2019.

[17] J. Zhang, X. Shi, I. King, and D.-Y. Yeung. Dynamic Key-Value Memory Networks for Knowledge Tracing. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 765–774, Perth, Australia, Apr. 2017. International World Wide Web Conferences Steering Committee.