Deep Learning or Deep Ignorance? Comparing Untrained Recurrent Models in Educational Contexts

Anthony F. Botelho[1], Ethan Prihar[2], Neil T. Heffernan[2]

[1] University of Florida, Gainesville FL 32611, USA
abotelho@coe.ufl.edu
[2] Worcester Polytechnic Institute, Worcester, MA, 01605
{ebprihar,nth}@wpi.edu

Abstract. The development and application of deep learning methodologies has grown within educational contexts in recent years. Perhaps attributable, in part, to the large amount of data that is made available through the adoption of computer-based learning systems in classrooms and larger-scale MOOC platforms, many educational researchers are leveraging a wide range of emerging deep learning approaches to study learning and student behavior in various capacities. Variations of recurrent neural networks, for example, have been used to not only predict learning outcomes but also to study sequential and temporal trends in student data; it is commonly believed that they are able to learn high-dimensional representations of learning and behavioral constructs over time, such as the evolution of a students' knowledge state while working through assigned content. Recent works, however, have started to dispute this belief, instead finding that it may be the model's complexity that leads to improved performance in many prediction tasks and that these methods may not inherently learn these temporal representations through model training. In this work, we explore these claims further in the context of detectors of student affect as well as expanding on existing work that explored benchmarks in knowledge tracing. Specifically, we observe how well trained models perform compared to deep learning networks where training is applied only to the output layer. While the highest results of prior works utilizing trained recurrent models are found to be superior, the application of our untrained-versions perform comparably well, outperforming even previous non-deep learning approaches.

Keywords: Deep Learning · LSTM · Echo State Network · Affect · Knowledge Tracing.

1 Introduction

The availability of large-scale education datasets, often comprised of large numbers of interactions between learners and educational technologies over time, have coincided with an increase in applications of deep learning methodologies

to study various aspects of student learning. The data collected from massive open online courses (MOOCs), for example, researchers have been able to utilize large, complex models to study student learning strategies as well as unproductive behavior such as attrition and dropout [6, 26, 21]. Even beyond MOOCs, in K-12 classrooms, the adoption of educational technologies and learning platforms such as Cognitive tutor [20] and ASSISTments [13], among many others, have led to the recording and often public release of large datasets of anonymized student interaction logs. The application of deep learning, collectively referring to a growing variety of multi-layer neural network models, often require large amounts of data to learn from, assuming, of course, that these models are in fact well-structured to learn anything at all.

Due to the large number of learned parameters and often complex structure of many deep learning approaches, many researchers and developers attribute the success of these methods to their ability to learn rich high-dimensional representations of input data. While it is possible to interpret and visualize what is learned in some applications of deep learning, it is difficult to what is learned within certain deep learning model structures including, for example, recurrent neural networks (RNN) [25]; this also includes commonly applied variants of RNN such as long short term memory (LSTM) [14] and gated recurrent unit (GRU) [8] networks. These model structures are designed to learn dependencies within time-series data, which is common in educational contexts.

Prior research that suggests, contrary to initial assumptions, that many recurrent models are not learning rich representations of data. In fact, it was found that by randomizing network weights and only training the output layer (referred to in this paper as "untrained" models), such models performed nearly as well as their trained counterparts [24, 9], as will be discussed further in the next section. This work seeks to build upon this prior research that has explored this phenomenon in the context of knowledge tracing [9], to compare trained and untrained recurrent models in another educational context, detecting student affect, where deep learning recurrent models have similarly been applied in recent years [3]. In addition, several related modeling approaches have been specifically designed to utilized randomized, untrained components. This work additionally explores the application of these approaches in educational contexts. Specifically, this work will address the following research questions:

1. How does the application of untrained recurrent models compare to similar trained models in detecting student affect?
2. Do the methods designed to utilize untrained components outperform other approaches in detecting student affect?
3. Do trained and untrained recurrent models exhibit an overlapping set of latent features within their hidden layers?

1.1 Representations within Recurrent Networks

While the application of recurrent networks, or one of several common variants, has increased in recent years, it is important to examine the basic structure of

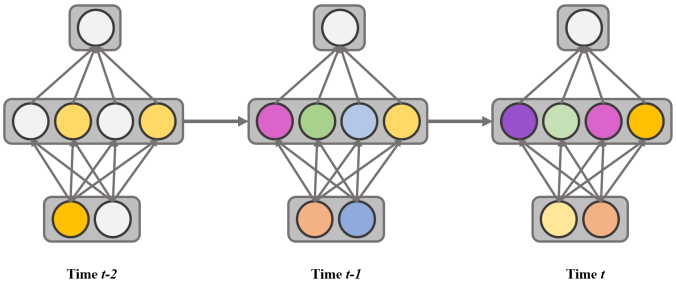

Fig. 1. This example illustrates how a recurrent network can build sequence representations within its hidden layer by combining new inputs with previous network states.

these networks in order to better understand how such models could learn rich knowledge representations. Consider, for example, the simplified network representation depicted in Figure 1. Like most "deep" models, recurrent networks are normally comprised of multiple layers of nodes representing values generated within the network structure. These values are calculated by multiplying the node values of earlier layers by learned weights that are traditionally fully-connected to all nodes in the subsequent layer. Unlike other network structures, recurrent models are designed to be applied to time-series data, where the values in the network's input layer (bottom layer in the figure) are combined with the previous hidden state (middle layer in the figure) for each time step. Intuitively, it is assumed that the model may learn how to combine new information with prior information within the series to make a more informed estimate for a given task; the network structure may learn how long to keep information, when to forget information, or certain conditions under which it should otherwise modify its understanding of the given sequence. The changing values within the hidden layer of the network contain and retain information from throughout the series.

How well these models are able to "understand" the given information, as represented by the values within the recurrent hidden layer, is a matter of recent speculation. It is precisely this question, as it applies to educational contexts, that is to be explored further within this work.

2 Background

In view of the application of recurrent models in education, it is important to better understand what applied models are learning from student data to understand how they can best be used to study various aspects of learning. Among the most well-known examples of using models to study learning, for example, is knowledge tracing [7]. In early models of knowledge tracing, interpretability was a primary goal; while later research has disputed how interpretable, or rather how identifiable traditional knowledge tracing models are [2, 10], the structure of the models were built in alignment to learning theory. While the models themselves

are trained by predicting short-term student performance across items within a given knowledge component, the goal of these models was to build a representation of student knowledge and learning. With the development and high reported performance of deep knowledge tracing (DKT) [19], questions were raised as to whether the recurrent neural network at its core was learning a more complex representation of student knowledge over time. Among a number of subsequent works that explored how "deep" this method truly was, Yeung et al. [28] showed that DKT's representation of student knowledge contradicted traditional learning theory as well as common sense; the model seemed to believe that students fluctuated frequently between states of knowledge and non-knowledge. While the authors proposed a fix to this problem using a form of regularization during model training, this work suggests that the model is able to perform well without a strong grasp of how learning is likely to occur.

These works, however, are not the first to question whether recurrent models are able to learn rich representations within sequential and temporal data. Wieting and Kiela [24] found that untrained recurrent models could perform comparably well to trained versions in natural language processing tasks. It was suggested that the applied models act as a type of sequence encoding, rather than by embedding deeper contextual information; the models are able to learn high-dimensional *encodings* of sequential data, but may be ignorant of the latent constructs and other factors that explain the data. The findings by Wieting and Kiela and subsequently Ding and Larson [9] who extended that work to further explore deep knowledge tracing, raise questions as to how useful these models are in educational contexts if they are not learning representations of deeper constructs; it is important to emphasize that these works did incorporate model training in the output layer of their compared models, so it is not the case that no training is required, only that the deeper layers of the networks may not learn composite features that align to latent factors in the data.

Prior work in applying recurrent models have explored how well such models are able to learn effective features in the context of detecting student affect and other learning behaviors [4]. In that work, the authors found that the use of expert-engineered features, developed in alignment to learning theory, led to higher predictive performance in a number of modeling tasks as compared to allowing a machine learning model to learn from the raw data logs used by the experts. The authors similarly identify an inherent difficulty within these networks to learn from the data.

This does not mean, however, that these models cannot still be useful in studying aspects of learning. Prior work has led to the development of sensor-free models of student affect [3] developed from student interaction logs paired with human-coded classroom observations [17, 18]. Utilizing LSTM networks, these models have been successfully applied to study student affect even without the ability to interpret the learned representations within the model [5]; even if it is the case that many recurrent models are unable to learn deep representations of latent constructs, this does not mean that the estimates produced by these models cannot be useful to study learning (c.f. "discovery with models" [22].

3 Methodology[3]

Although there are many advantages in utilizing interpretable models to study learning, there are several practical benefits made possible if recurrent models are truly "ignorant" to deeper representations within data. As has been found in the prior works described in the previous section (i.e. [24, 9]), untrained variants of recurrent models may perform comparably to similar trained models in several applications. If this finding holds in other contexts, these models may have increased potential for integration in a number of educational technologies and settings by significantly reducing training times or potentially even the amount of data needed to successfully fit such a model (needing to train only the output layer would be equivalent, in most cases, to training a linear or logistic regression model which traditionally requires fewer data samples to train).

In this work, we use student affect detection and knowledge tracing as two example cases of comparison for trained and untrained recurrent models on previously-published benchmark results ([4] and [27] for affect detection and knowledge tracing, respectively). While Ding and Larson [9] did explore applications of untrained models in the context of knowledge tracing, this work further expands upon this work by introducing two additional methods, Bag of Random Embeddings [24] and Echo State Networks [24, 15], within educational contexts.

3.1 Affect and Knowledge Tracing Data

In this paper, we observe applications of these trained and untrained recurrent network models within two publicly-available datasets collected within ASSISTments [13]. ASSISTments is a free web-based learning platform used by primarily middle-school teachers and students for mathematics homework and classwork. Among a number of other features, the learning system allows teachers to assign traditional "complete all problems" assignments as well as mastery-based "skill builder" assignments. While working through problems, the system allows students to make multiple attempts to answer problems and offers supports in the form of on-demand hints and scaffolding problems. To support educational research, the system has also released a number of publicly-available datasets such as those utilized within this paper.

The first dataset observed in this work was released in [4], which was derived from several prior works focused on the development of sensor-free detectors of student affect (e.g. detectors that utilize only interaction logs without additional sensors such as video). ASSISTments data was used to develop affect detectors using expert-engineered features based on both theory and an iterative development process [18]. Additional works subsequently experimented with different features within a number of rule- and regression-based modeling methods [23] before recurrent deep learning methods were explored [3].

The dataset itself is comprised of student interaction logs paired with human-coded classroom observations of four states of student affect: engaged concentration, boredom, confusion and frustration. Following [4], the data exists in

[3] The code utilized by this work is made publicly available: https://osf.io/ubr2v/

two forms, the first consisting of the 92 expert-engineered features used in prior works, and second consisting of the raw action-level logs that were used to build these features. While that work found that the use of expert-engineered features led to superior model performance as compared to the raw features, we explore both feature sets in this work utilizing untrained models to examine the performance benefits of training these models.

The second dataset observed in this work has been previously used to examine methods of knowledge tracing [27]. As described in Section 2, knowledge tracing is among the most widely studied problems in learner analytics, AI in education, and educational data mining communities. The original knowledge tracing (KT) model [7] and its bayesian implementation (BKT), attempt to model student latent knowledge using student performance metrics. The ASSISTments knowledge tracing dataset used in this work was made publicly available in [27] (specifically, the dataset refered to as "09-10 (c)" in that paper), after fixing several identified errors in the original version of that dataset used in Piech et al.'s original deep knowledge tracing paper [19]. This dataset is comprised of 275,459 math problems across 146 knowledge components answered by 4,217 students.

3.2 Leveraging Untrained Networks

This work explores the application of several untrained model structures. These model structures were applied across both the affect and knowledge tracing datasets, predicting the affect labels (as a multi-dimensional categorical outcome, as was done in prior works) and next problem correctness, respectively. As previously introduced, the terminology of "untrained" in the context of this work (in alignment with prior works [24, 9]), refers to a partially-trained model. In most machine learning contexts, especially those observing deep learning approaches, models are typically trained by randomizing the initial values of a set of weights or coefficients that are then updated iteratively through an optimization procedure [16]. Considering deep learning models, this process is believed to help the model learn sets of features in lower layers of the network, with the final output layer (often functionally equivalent to a linear or logistic regression) then learning how to map those features to a set of outcomes. An "untrained" method effectively skips the optimization procedure, relying on the randomized weights to produce a large number of un-tuned features; in this process, a single regression model can be trained using these un-tuned features to map them to observed outcomes. This is an important distinction as this creates somewhat of a misnomer in that these methods still rely on some degree of training, but do not rely on training to "model" the data. These methods, as well as the application procedure, is described in this section.

Bag of Random Embeddings The bag of random embeddings was the simplest untrained network. This method is used to simply project the time series data to a higher dimensional space. To create a bag of random embeddings for a time series of f features and t time-steps, the approach projects the time series

into a n dimensional embedding by first creating a t by f matrix, referred to as the time-series matrix, where each row in the matrix is the full set of features from one time-step. Next, the approach generates an f by n matrix full of random values, referred to as the projection matrix. The time-series matrix is then multiplied by the projection matrix, resulting in a t by n matrix, referred to as the embedding matrix. Finally, a pooling operation is applied across all the time-steps in the embedding matrix, resulting in a final n dimensional vectorized embedding of the initial time series.

Following the advice of [24], the random numbers of the projection matrix were initialized between $\frac{-1}{\sqrt{f}}$ and $\frac{1}{\sqrt{f}}$. To find the best bag of random embeddings, all combinations of an n of 1, 2, 4, 8, 16, 32, 64, 128, 256, and 512, and both max and mean pooling, for five random seeds each were used to project the time-series data before using a 5-fold cross-validated logistic regression to classify affect or predict next problem correctness, depending on the dataset. The average performance of every fold of every random seed for each combination of hyper-parameters was used to determine the best values.

Long Short-Term Memory Networks The Long Short-Term Memory Network (LSTM) [14, 11] is a common recurrent network structure for modeling time-series data. The LSTM network is a form of recurrent neural network that in addition to utilizing information from its past state, is designed to learn when to incorporate new information into its state and when to forget previous information. In this context, the value of the LSTM network is often viewed as being in its internal state structure which incorporates a type of memory that is designed to capture long- and short-term dependencies within the series (thus its name). Even without training, the state of the LSTM network, if complex enough, can capture useful, predictive information from the time-series in certain contexts [24]. To determine if an untrained LSTM network would be capable of predicting either affect or next problem correctness, an LSTM network was created with all combinations of zero through four hidden layers (i.e. additional fully connected layers on top of the LSTM layer), and 1, 2, 4, 8, 16, 32, 64, 128, 256, and 512 nodes per layer, including the output layer, for five random seeds. Each network's output layer was given to a 5-fold cross-validated logistic regression and used to classify affect or predict next problem correctness. The average performance of every fold of every random seed for each combination of hyper-parameters was used to determine the best combination of hyper-parameters.

Echo State Networks Echo State networks are similar to recurrent networks in that they have connections from forward nodes to their predecessors, but these networks usually lack the formality of layers. Instead, an echo state network has a reservoir of nodes that have many connections to many other nodes in the reservoir. The input layer connects to any subset, or all of the nodes in the reservoir, and the output layer receives the output from the reservoir nodes. The weights in the reservoir are never trained, but the weights of the output layer are

Table 1. Comparison of trained and untrained models applied to the affect dataset

Model	Features	Best Model	AUC	Kappa
Untrained Models				
LSTM Network	Raw	$n = 512$, 0 *added hidden*	0.661	0.098
Bag of Random Emb.	Raw	$n = 64$, *max pooling*	0.631	0.066
Echo State Network	Raw	$n = 512$, 1 *added hidden*	0.673	0.121
LSTM Network	Expert	$n = 512$, 1 *added hidden*	0.701	0.152
Bag of Random Emb.	Expert	$n = 64$, *mean pooling*	0.741	0.128
Echo State Network	Expert	$n = 512$, 0 *added hidden*	0.694	0.127
Trained Models				
LSTM (Botelho et al., 2019)	Raw		0.695	0.041
LSTM (Botelho et al., 2019)	Expert		0.760	0.172

[15]. The echo state network is designed to exploit the properties of a recurrent network's state similarly to how the previous section uses the state of an LSTM network. Within the untrained weights of the reservoir lies the state of the echo state network. This state is designed to capture the latent information of the time-series data presented to it and when the output layer is trained.

To determine if an echo state network would be capable of predicting either affect or next problem correctness, the output of each of the random LSTM networks from the previous section was combined with the intermediate output from every node in the network, essentially converting the LSTM network to an echo state network. The outputs from every node were again used to classify affect or predict next problem correctness in a logistic regression, which functions as the output layer of the echo state network. The average performance of every fold of every random seed for each combination of hyper-parameters was used to determine the best combination of hyper-parameters.

4 Results

The results of our applied untrained models are compared to the results generated from trained models as reported in prior works utilizing the same respective datasets used here. For consistency, these results are compared using the same metrics as have been used in comparison in prior works; in regard to the affect data, the AUC measure is calculated using the multi-class categorical evaluation method as used in previous works [12].

The results of the untrained models applied in this work in comparison to the trained models described in [4] are reported in Table 1. The highest-performing of each model type is compared to the reported results of the prior work across measures of AUC and Kappa (in alignment to that prior work). In this table, it can be seen that the trained LSTM utilizing expert-engineered features exhibits the highest model performance across both metrics. However, the untrained LSTM and Bag of Random Embedding models each perform comparably close in regard to AUC and Kappa; these even outperform the trained LSTM model utilizing the raw dataset.

Table 2. Comparison of trained and untrained knowledge tracing models.

Model	Best Model	AUC
Untrained Models		
LSTM Network	$n = 512$, 0 *added hidden*	0.706
Bag of Random Emb.	$n = 512$, *mean pooling*	0.692
Echo State Network	$n = 512$, 0 *added hidden*	0.725
LSTM (Ding & Larson, 2019)		0.730
Trained Models		
DKT (LSTM; Xiong et al., 2016)		0.750
BKT (Xiong et al., 2016)		0.630

Similarly, the results of the untrained models applied in this work in comparison to previous results are reported in Table 2. In this table, we also compare our untrained model results to the untrained model applied in [9]. Here, the trained DKT model does exhibit the highest AUC performance, but the untrained LSTM as reported in [9] and Echo State Network applied in this study perform comparably well. What is perhaps particularly worth noting, is that all untrained recurrent models outperformed the BKT model.

5 Exploring Latent Feature Overlap

We have seen over the previous set of analyses that the untrained models perform comparably well to their trained counterparts. This raises several questions including what, if anything, is being learned within the hidden layer of these trained recurrent models (i.e. is there an overlap of latent features utilized by these models). In addressing our third research question, we conduct a final analysis to explore the latent features represented by trained and untrained models in detecting student affect.

In this analysis, we compare an LSTM-based model architecture as presented in [3] as a basis of comparison. We train this method using one LSTM layer consisting of 200 nodes feeding to a dense output layer of 4 nodes corresponding to the four affective states, similar to those previously described. We train this model and then extract the hidden layer from the network. Similarly, we generate five untrained counterparts using the same model architecture differing only in the number of nodes used in the hidden layer (using 200, 400, 600, 800, and 1000). We similarly extract the hidden layers of these models corresponding with each sample of the affect detection dataset.

We conduct an exploratory factor analysis (EFA) to identify latent constructs represented by each set of hidden features. EFA is a common dimensionality reduction method that identifies latent factors, or features, that exist as the linear combination of other features [1]. With this, we want to observe whether the factors that emerge from the trained model overlap, or are meaingfully correlated, with the untrained model factors. If the trained model is not learning effectively from the data, we would expect that there would be a large overlap in factors when compared with the untrained models.

Table 3. Number of factors and overlap between untrained and trained models.

Model	EFA Factors	N Overlapping Factors (Rho>0.6)
Trained LSTM (200)	31	—
Untrained LSTM (200)	35	5
Untrained LSTM (400)	50	1
Untrained LSTM (600)	74	5
Untrained LSTM (800)	91	4
Untrained LSTM (1000)	103	5

From our EFA, reported in Table 3, 31 features emerge from our trained model, with an increasing number of factors emerging from larger untrained dimensions (the number of factors were determined based on the number of factors with an eigenvalue greater than 1, following common practice). Using these features, we conducted a complete pair-wise comparison of untrained factors to trained model factors and computed a Spearman (Rho) ranked correlation for each pairing. We then simply counted the number of factor pairs that exhibited a Rho value greater than 0.6 as a measure of pseudo-overlapping feature sets. From the table, it can be seen that despite the increasing number of emerging factors, the number of overlapping factors remained relatively constant. This suggests that, while the untrained models constructed large feature sets, these were mostly uncorrelated with the trained model features.

6 Discussion

While it is surprising that the untrained models perform comparably well to their trained counterparts, the results of our analyses suggest that the trained models are learning effectively from the data; particularly from the EFA, we argue that the learned features are not simply random combinations of features due to the notable lack of overlap with the factors emerging from the untrained models. This lack of overlap is unexpected given the comparable model performance, suggesting that there are a small number of highly-predictive factors present.

In both affective and knowledge tracing contexts, the untrained models perform remarkably well, even outperforming other benchmarks (e.g. the trained LSTM using the raw data in Table 1 and the BKT model in Table 2). This work represents a step toward better understanding how deep learning models learn from given data. It is difficult to conclude that our findings will generalize to *all* recurrent models and applications, but the analyses conducted in this work in conjunction with those presented in prior works [24, 9] have found similar results across multiple contexts. It is the goal that this work will lead to further work to better understand knowledge representations within deep learning models to either better utilize them in various contexts, or to improve them so that they may exhibit higher utility for the study of learning.

Following the results reported in this paper, it is important to clarify and emphasize the contribution and potential impact of our findings. First, as the

untrained models were found to be comparable to prior results across both applications observed in this paper, this finding aligns with prior research that suggests that the trained recurrent models may not be learning deep representations. However, the lack of overlap between factors emerging from the trained and untrained models suggests that the trained model is learning a distinctive set of latent factors related to affect. This finding supports the use of such models to both detect affect, but also to better study the latent structures that indicate affect and other learning constructs (i.e. these features are not simply randomly generated or encoded features). With that said, the untrained models may additionally provide utility. As the models perform well above chance and other simple baselines, the estimates produced by these models may still highly correlate with outcomes of interest and may be used to study learning.

7 Acknowledgements

We would like to thank NSF (e.g., 2118725, 2118904, 1950683, 1917808, 1931523, 1940236, 1917713, 1903304, 1822830, 1759229, 1724889, 1636782, & 1535428), IES (e.g., R305N210049, R305D210031, R305A170137, R305A170243, R305A180401, & R305A120125), GAANN (e.g., P200A180088 & P200A150306), EIR (U411B190024), ONR (N00014-18-1-2768) and Schmidt Futures.

References

1. Bandalos, D.L., Finney, S.J.: Factor analysis: Exploratory and confirmatory. In: The reviewer's guide to quantitative methods in the social sciences, pp. 98–122. Routledge (2018)
2. Beck, J.E., Chang, K.m.: Identifiability: A fundamental problem of student modeling. In: International Conference on User Modeling. pp. 137–146. Springer (2007)
3. Botelho, A.F., Baker, R.S., Heffernan, N.T.: Improving sensor-free affect detection using deep learning. In: International Conference on Artificial Intelligence in Education. pp. 40–51. Springer (2017)
4. Botelho, A.F., Baker, R.S., Heffernan, N.T.: Machine-learned or expert-engineered features? exploring feature engineering methods in detectors of student behavior and affect. In: The 12th international conference on educational data mining (2019)
5. Botelho, A.F., Baker, R.S., Ocumpaugh, J., Heffernan, N.T.: Studying affect dynamics and chronometry using sensor-free detectors. International Educational Data Mining Society (2018)
6. Chaplot, D.S., Rhim, E., Kim, J.: Predicting student attrition in moocs using sentiment analysis and neural networks. In: AIED Workshops. pp. 54–57 (2015)
7. Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction **4**(4), 253–278 (1994)
8. Dey, R., Salem, F.M.: Gate-variants of gated recurrent unit (gru) neural networks. In: 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS). pp. 1597–1600. IEEE (2017)
9. Ding, X., Larson, E.C.: Why deep knowledge tracing has less depth than anticipated. International Educational Data Mining Society (2019)

10. Doroudi, S., Brunskill, E.: The misidentified identifiability problem of bayesian knowledge tracing. International Educational Data Mining Society (2017)
11. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: Continual prediction with lstm (1999)
12. Hand, D.J., Till, R.J.: A simple generalisation of the area under the roc curve for multiple class classification problems. Machine learning **45**(2), 171–186 (2001)
13. Heffernan, N.T., Heffernan, C.L.: The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. International Journal of Artificial Intelligence in Education **24**(4), 470–497 (2014)
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
15. Jaeger, H.: Echo state network. scholarpedia **2**(9), 2330 (2007)
16. Le, Q.V., Ngiam, J., Coates, A., Lahiri, A., Prochnow, B., Ng, A.Y.: On optimization methods for deep learning. In: ICML (2011)
17. Ocumpaugh, J.: Baker rodrigo ocumpaugh monitoring protocol (bromp) 2.0 technical and training manual. Tech. rep., Technical Report. New York, NY: Teachers College, Columbia University ... (2015)
18. Ocumpaugh, J., Baker, R., Gowda, S., Heffernan, N., Heffernan, C.: Population validity for educational data mining models: A case study in affect detection. British Journal of Educational Technology **45**(3), 487–501 (2014)
19. Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L., Sohl-Dickstein, J.: Deep knowledge tracing. In: Proceedings of the 28th International Conference on Neural Information Processing Systems. pp. 505–513 (2015)
20. Ritter, S., Anderson, J.R., Koedinger, K.R., Corbett, A.: Cognitive tutor: Applied research in mathematics education. Psychonomic bulletin & review **14**(2), 249–255 (2007)
21. Rosé, C.P., Carlson, R., Yang, D., Wen, M., Resnick, L., Goldman, P., Sherer, J.: Social factors that contribute to attrition in moocs. In: Proceedings of the 1st ACM conference on Learning@ scale conference. pp. 197–198 (2014)
22. Siemens, G., Baker, R.S.d.: Learning analytics and educational data mining: towards communication and collaboration. In: Proceedings of the 2nd international conference on learning analytics and knowledge. pp. 252–254 (2012)
23. Wang, Y., Heffernan, N.T., Heffernan, C.: Towards better affect detectors: effect of missing skills, class features and common wrong answers. In: Proceedings of the 5th International Conference on Learning Analytics and Knowledge. pp. 31–35. ACM (2015)
24. Wieting, J., Kiela, D.: No training required: Exploring random encoders for sentence classification. arXiv preprint arXiv:1901.10444 (2019)
25. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural computation **1**(2), 270–280 (1989)
26. Xing, W., Chen, X., Stein, J., Marcinkowski, M.: Temporal predication of dropouts in moocs: Reaching the low hanging fruit through stacking generalization. Computers in human behavior **58**, 119–129 (2016)
27. Xiong, X., Zhao, S., Van Inwegen, E.G., Beck, J.E.: Going deeper with deep knowledge tracing. International Educational Data Mining Society (2016)
28. Yeung, C.K., Yeung, D.Y.: Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In: Proceedings of the 5th Annual ACM Conference on Learning at Scale. pp. 1–10 (2018)