Improving Sensor-Free Affect Detection Using Deep Learning

Anthony F. Botelho[1], Ryan S. Baker[2], and Neil T. Heffernan[1]

[1] Worcester Polytechnic Institute, Worcester, MA
{abotelho,nth}@wpi.edu,
[2] Teachers College, Columbia University, New York, NY
ryanshaunbaker@gmail.com

Abstract. Affect detection has become a prominent area in student modeling in the last decade and considerable progress has been made in developing effective models. Many of the most successful models have leveraged physical and physiological sensors to accomplish this. While successful, such systems are difficult to deploy at scale due to economic and political constraints, limiting the utility of their application. Examples of "sensor-free" affect detectors that assess students based solely using data on the interaction between students and computer-based learning platforms exist, but these detectors generally have not reached high enough levels of quality to justify their use in real-time interventions. However, the classification algorithms used in these previous sensor-free detectors have not taken full advantage of the newest methods emerging in the field. The use of deep learning algorithms, such as recurrent neural networks (RNNs), have been applied to a range of other domains including pattern recognition and natural language processing with success, but have only recently been attempted in educational contexts. In this work, we construct new "deep" sensor-free affect detectors and report significant improvements over previously reported models.

Keywords: Deep Learning; Affect; Sensor-Free; Recurrent Neural Networks; Educational Data Mining;

1 Introduction

While intelligent tutors have a long history of development and use, the most widely-used systems remain less sophisticated than initial visions for how they would operate. The systems now used at scale are often cost-effective and have been shown in large-scale randomized controlled trials to lead to better learning outcomes (e.g. [1],[2]), but do not reach the full level of interactivity of which human tutors are capable. For example, one positive aspect of human tutors is the ability to observe student affective state and adjust teaching strategies if students are exhibiting disengaged behavior [3]. Student emotion and affective state have been found to correlate with academic performance [4][5] and can even be used to predict which students will attend college [6].

With increasing evidence supporting the benefits of utilizing student affective state to drive tutoring strategies [7], it is important to develop accurate means of detecting these states from students working in these systems. While strides have been made to build accurate detectors, many successful approaches include the use of physical and physiological sensors [8][7][9]. However, it can be impractical to deploy such sensors to classrooms at scale, both for political and financial reasons. Detecting affect solely from the interaction between the student and learning system, sometimes referred to as sensor-free affect detection, may be more feasible to deploy at scale. However, while these models' predictions have been usable in aggregate for scientific discovery, the goodness of these approaches has often been insufficient for use in real-world intervention.

Sensor-free affect detectors have existed for several years and have been used to assess student affective states using low-level student data as students interact with a mouse and keyboard [10] , but also using features extracted from a range of learning platforms including Cognitive Tutor [11], AutoTutor [12], Crystal Island [13], and ASSISTments [14][15]. While these detectors have been better than chance, their goodness has fallen short of detectors of disengaged behavior, for example (cf. [5]). Increasing the accuracy of sensor-free affect detectors would lead to higher confidence in their use to drive intervention.

In this paper, we attempt to enhance sensor-free affect detection through the use of "deep learning," or specifically, recurrent neural networks (RNNs) [16]. Previous affect detectors have utilized a range of algorithms to detect student affective state; we study whether deep learning can produce better predictive accuracy than those prior algorithms. We study this possibility within a previously published data set to facilitate comparison with and understanding of the benefit derived from using this algorithm. Recurrent neural networks are a type of deep learning neural network that incorporates at least one hidden layer, but also provides an internal hidden node structure that captures recurrent information in time series data.

RNNs are most appropriately applied to time series data, where the output of the current time step is believed to be influenced or impacted by previous time steps. In this way, it is believed that affect detection could benefit from a model that observes the temporal structure of input data. Several internal node structures have been proposed, yielding variants of traditional RNNs such as Long-Short Term Memory networks (LSTMs) [17] and more recently Gated Recurrent Unit networks (GRUs) [18]. Applications of these deep learning algorithms have been used in other domains for pattern recognition [19] and improving natural language processing [20]. Performance in these domains certainly suggest large benefits in using deep learning on temporal or time series information.

Deep learning prediction models have not yet been used extensively in educational domains, but have been studied as a potential method to improve the decisions of virtual agents in game-based learning environments [21] and also to improve the prediction of student correctness on the next problem [22]. However, the results of the "Deep Knowledge Tracing" (DKT) model presented in [22] are as yet uncertain; initial reports suggested profoundly better performance than previous approaches, but later investigation by other researchers indicated that the same data points were being replicated and used to predict themselves, artificially inflating goodness [23]. When this error was corrected, performance seemed to be equivalent to previous approaches [24]. Nonetheless, recurrent neural networks may be highly effective for problems with the complexity and the quantity of data available to fully leverage their benefits.

As such, this work seeks to apply deep learning to utilize student information to better detect students' affective states without the use of sensors. We explore the application of recurrent neural networks for the task of detecting affective states using data collected in the context of the ASSISTments online learning platform.

2 Dataset

The dataset[1] used to evaluate our proposed deep learning approach to detecting affective state is drawn from the ASSISTments learning platform [25]. ASSISTments is a free web-based platform that is centered around providing immediate feedback to the many students who use it in the classroom and for homework daily. ASSISTments also provides on-demand hints and sequences of scaffolding support when students make errors. The system was used by over 40,000 students across nearly 1,400 teachers during the 2015-2016 school year, and has been found to be effective in a large-scale randomized controlled trial [2].

2.1 Data Collection and Feature Distillation

The ground truth labels used in this dataset come from in-class human observations conducted using the Baker-Rodrigo Ocumpaugh Monitoring Protocol (BROMP) [26]. These quantitative field observations (QFOs) were made by trained human coders who observed students using the ASSISTments learning platform in a classroom environment. The coders observed students and labeled their affect as bored, frustrated, confused, engaged concentration, or other/impossible to code. They collected affect observations over 20-second intervals in a round-robin fashion, cycling through the entire class between observations of a specific student. Unlike approaches using video coding or retrospective emote-aloud (e.g. [27]), this approach inherently leads to missing labels between observations of the same student. These missing intervals for each student are known, as timestamps are recorded for each observation, and will be taken into account when formatting the data for input into the recurrent neural network; this process is described in more detail in a later section.

A total of 7,663 field observations were obtained from 646 students in six schools in urban, suburban, and rural settings. In prior work [15], a set of 51 action-level features was developed using an extensive feature engineering process; these features consist of within- and across-problem behaviors including response behavior, time working within the system, hint and scaffold usage within

[1] Our dataset is made available at http://tiny.cc/affectdata

the system, and other such features attempting to capture various low-level student interactions with the system. As the observation intervals, or clips, often contain more than one student action within the learning system, the features were aggregated within each clip by taking the average, min, max, and sum of each feature. The end result was 204 features per clip.

In this paper we will compare our deep learning-based detectors of student affect to two earlier sensor-free models of student affect within ASSISTments (e.g. [14][15]). In doing so, we will use the exact same training labels and features as in [15], in order to focus our comparison solely on the use of deep learning.

3 Methodology

We input these labels and features into three deep learning models representing three common variants of recurrent networks including a traditional recurrent neural network (RNN), a Gated Recurrent Unit (GRU) neural network, and a Long-Short Term Memory network (LSTM). The GRU variant was chosen when exploring network structures and hyperparameters for training for both its faster training times in comparison to the LSTM variant and also for its increased ability to avoid problems such as vanishing gradients to which traditional RNNs are more susceptible. The models explored in this work were built in python using the Theano [28] and Lasagne [29] libraries.

3.1 Network Structure

Our implementations each use the same three layer design, with an input layer feeding into a hidden recurrent layer of 200 nodes, progressing to an output layer of four nodes corresponding to each of four classes of affective state. The input layer accepts a student-feature vector of 204 generated covariates per time step normalized using the mean and standard deviation of the training set, and each network ultimately outputs 4 values representing the network's confidence that the input matches each of the four labels of engaged concentration, boredom, confusion, and frustration. A rectified nonlinear activation function is used on the output of the hidden layer, while a softmax activation function is used for the final model output.

Due to the large number of parameters present in deep learning networks, it is common to implement techniques to avoid overfitting. We adopt the common practice of incorporating dropout [30] into our model, which, in a general sense, sets some network weights to 0 with a given probability during each training step. This creates a changing network structure in terms of its interconnectivity during training to help prevent the model from relying on just a small number of input values. In our three layer model, dropout can be applied before and/or after the recurrent layer, and this is explored to determine which location of placement produces superior performance. We incorporate 30% dropout, such that each weight in the network, in the location dropout is applied, has a 30% chance of being dropped for a single training step; many implementations instead describe dropout in terms of a “keep” probability, but is described here as a “drop” probability to remain consistent with the library used to build the models. As is standard practice, dropout is not used when applying the model to the test set.

3.2 Handling Time Series Data and Labels

The dataset used for the previous detectors in ASSISTments, and again in this work, consists of 20 second interval clips to which an affect label has been applied. The recurrent network takes as input a sequence of these clips to make use of the recurrent information within the sequence. The labeled clips, however, are not consecutive due to the design of the field observations, leading to gaps in student observations; during a gap in one student’s sequence, the human coders present in the classroom were observing other students. It is possible to represent the non-consecutive clips as a full sequence, however, treating clips that are distant in time as consecutive may confuse the network and reduce performance. For this reason, we treat clips as consecutive only if they occur within 5 minutes of the previous labeled clip. Clips that occur beyond this threshold form a new sequence sample, resulting in a larger number of samples consisting of shorter sequences.

Another issue presented by the classification task is the non-uniformity of the distribution of the labels. The vast majority, approximately 80% of the clips, are labeled as engaged concentration, followed by 12% labeled as boredom, and only 4% each of confusion and frustration. While it is perhaps encouraging to know that students are mostly concentrating when working within ASSISTments, a model trained with labels in such non-uniformity may bias in favor of the more frequent labels. While it is often beneficial for the model to understand this distribution to some extent, it is better for the model to learn the trends in the data that correspond to each label rather than simply learn the overall distribution.

The original, non-recurrent affect detectors corrected for this issue by resampling each of the labels [5], but this cannot be directly reproduced here due to the time-series input into the recurrent network. In that previous work, the training data was sampled with replacement proportional to the distribution such that the resulting dataset is balanced across the distribution of labels and then evaluating on a non-resampled test set [31]. Rather than representing each sample as independent as in previous detectors, the recurrent network observes a sequence of observations within a single training sample. As such, we resample entire sequences including rarer affective states. Resampling in this way is likely to also resample the other labels as well, particularly when resampling the more scarce labels of frustration and confusion. While it is difficult to achieve perfect uniformity, sampling with replacement is performed using a threshold to balance the labels to a feasible degree. In this way, each sample of the training set is selected at least once, duplicating only those sequences containing at least 20% of one of the less common labels. From the resulting resampled data, we randomly downsample to the size of the original non-resampled training set for faster training times; training on the full resampled dataset did not produce substantial gains in model goodness over using the downsampled training set.

In an effort to further account for the non-uniformity of the distribution of labels, a final normalization is applied to the output of the network. The training data is used to determine the minimum and maximum prediction values for each label that is then used to scale the resulting predictions during model evaluation to span the entire 0 to 1 range (any prediction values in the test set outside of this range are truncated). This rescaling helps to deter the model from making overly conservative estimates of the less frequent labels. The output normalization is found to be necessary in this regard as estimates for the scarce labels rarely surpassed a 0.5 rounding threshold after the softmax activation of the output.

3.3 Model Training

All models are evaluated using 5-fold cross validation, split at the student level to evaluate how the model performs for unseen students. It is often common, in working with neural networks, to train using mini-batches of samples, updating model weights based on the outputs over several training steps. In the case of recurrent neural networks, the data contains multiple time steps that the model treats as a batch and updates the network weights at the end of the sequence. We update the model after each sample sequence using an adaptive gradient descent calculation [32], and categorical cross-entropy is used as the cost function for model training due to its ability to handle multi-label classification; each sample contains a varying number of individual time steps, over which the network makes a single update from the aggregated cost.

Each model is trained over a multitude of epochs, or full cycles through the training set. Training over too many epochs or too few can reduce performance through overfitting and underfitting respectively. The appropriate number of epochs will also differ when applying models of different complexities, as is being done in this work. For this reason, we hold out 20% of each training set as a validation set and incorporate an “early stop” criterion for model training. After each epoch the model evaluates its performance on the unseen validation set to determine the point in training where there is little or no improvement.

A moving average of the model’s error on the validation set, expressed as average cross-entropy (ACE) for training, is calculated over the most recent 10 epochs (starting with the 11th epoch). The model stops training when it finds that moving average value at a particular epoch is larger than or equal to the previously calculated average (lower values indicate superior ACE values). Using this criterion allows for a more fair comparison of the performance of each model. Although a maximum number of 100 epochs was allowed, no models in this paper reached that maximum threshold.

4 Measures

We will evaluate the results of each of our model evaluations through three statistics, AUC ROC/A’, Cohen’s kappa, and Fleiss’ kappa. Each kappa uses a

Model	AUC	Cohen's Kappa	Fleiss' Kappa
30% Dropout Before Recurrent Layer	0.74	0.12	0.22
30% Dropout After Recurrent Layer	0.74	0.13	0.23
30% Dropout Before & After Recurrent Layer	0.73	0.11	0.21

Table 1. Comparing locations of dropout within the GRU model.

0.5 rounding threshold. This is a multi-label classification task such that each sample has one of four possible labels of confusion, concentration, boredom, or confusion. For this reason, the metrics of AUC and Cohen's kappa are first calculated for each of the four labels independently, and the final result is an average across the four labels [33]. It is not common to report average Cohen's kappa for multi-label classification; we include this metric for comparison to previous results reporting this metric. We also report Fleiss' kappa, which is better suited for multi-label classification, taking all label comparisons into account in a single metric. Both kappa metrics are reported as secondary measures, as AUC is unaffected by scaling and rounding threshold-setting procedures. In all cases, we report performance on the test data, averaged across each fold of a 5-fold cross validation.

5 Results

5.1 Adjusting the Dropout Context

Our initial analysis pertains to the degree of impact the context of dropout has on model goodness. We investigate this question in the context of the GRU model and the resampled training dataset, looking at whether dropout occurs before the recurrent layer, after the recurrent layer, or both. In all cases, a 30% hyperparameter is used for the dropout percentage. Table 1 shows that when dropout occurs has little impact on performance. When dropout is applied to both areas of the model, however, there is a mild reduction in both metrics, suggesting that applying dropout in both locations impedes model training to a noticeable degree. For this reason, all further models reported used dropout applied after the recurrent layer. This placement is chosen as there is a very slight increase in both Cohen's and Fleiss' kappa; additionally, it is more common for researchers and practitioners to apply dropout after the recurrent layer.

5.2 Comparing RNN Variants

We next compare a traditional recurrent neural network (RNN), a Gated Recurrent Unit (GRU) network, and a Long-Short Term Memory network (LSTM), which vary in their complexity, and as such in their number of parameters and

Model	AUC	Cohen's Kappa	Fleiss' Kappa
RNN With Resampling	0.73	0.14	0.22
GRU With Resampling	0.74	0.13	0.23
LSTM With Resampling	0.73	0.11	0.22
RNN Without Resampling	**0.78**	0.19	0.24
GRU Without Resampling	0.77	0.19	0.24
LSTM Without Resampling	0.77	**0.21**	**0.27**
Wang et al. [15]	0.66	0.25	–
Ocumpaugh et al. [14]	0.65	0.24	–

Table 2. Three recurrent model variants, trained on both the resampled and non-resampled datasets, are compared to the previous highest reported results on the ASSISTments dataset.

flexibility of fit. These models are compared using the same training and test data sets and differ only in the internal node structure used for the network. In parallel, we examine the effects of adjusting the training data (but not the test data) using resampling, by comparing each model variant trained on the resampled dataset to that model variant trained on a data set without resampling.

The performance of each model is compared in Table 2. In all three model variants, training on the non-resampled data produced superior performance in all metrics over training with the resampled data, contrary to our initial hypothesis. Also contrary to our initial hypothesis, the GRU models did not produce the best outcomes; instead, the simplest model, the traditional RNN, was found to have superior AUC performance to the other models, albeit only by a small margin. This may be because it had the fewest parameters; the RNN trains approximately 82,000 parameters as compared to the over 244,000 parameters in the GRU model and nearly 326,000 parameters in the LSTM model. This smaller number of parameters also leads to the RNN being the fastest model to train. The LSTM model, however, had higher kappa values than the other network variants, and as such, could also be argued to be the best model as it exhibits comparably high AUC values and also would be able to handle longer sequences than a traditional RNN if used in real-time applications. All three deep learning models achieve substantially better AUC than the best models produced through prior work using more traditional machine learning algorithms (e.g. [14][15]). Cohen's kappa, however, is found to be slightly worse than in the prior efforts.

Performance was generally good for AUC across all affective states, as shown in Table 3. It becomes apparent, however, that performance is not well-balanced across the labels. The difference between AUC and kappa values suggests that the model for confusion, for example, is generally able to distinguish between

	Resampled		Non-Resampled	
	AUC	Cohen's Kappa	AUC	Cohen's Kappa
Confused	0.67	-0.01	0.72	0.09
Concentrating	0.78	0.24	0.80	0.34
Bored	0.76	0.18	0.80	0.28
Frustrated	0.68	0.01	0.76	0.15
Average	0.73	0.11	0.77	0.21

Table 3. LSTM model performance for each individual affect label.

confused and non-confused students, but is poor at selecting a single threshold for this differentiation. The difference between affective states is likely associated with their relative frequency; the best-detected affective states (concentrating and boredom) were also the most common ones. While resampling was chosen to address this problem, Table 3 also shows that this technique, as implemented, did not lead to better performance.

6 Discussion and Future Work

Despite their broad application in other domains, deep learning models have been relatively under-utilized in education and their application often has not led to better results than other common algorithms [24]. In this paper, we attempt to apply deep learning to the problem of sensor-free affect detection, using a data set previously studied using more traditional machine learning algorithms. Three deep learning models (RNN, GRU, and LSTM) were compared to previously published work. All three deep learning models explored here obtained substantially better AUC than past results reported using the same dataset, although they did not lead to better values of Kappa. This difference between metrics is not surprising, given that the cost function implemented in the deep learning models does not round each prediction before evaluating each class label, but instead evaluates the degree of error across all classes each training step. Nonetheless, the substantially higher AUC values argue that deep learning models may prove a very useful tool for research and practice in sensor-free affect detection, eventually leading to models that can be more effectively used both to promote basic discovery and to drive affect-sensitive intervention.

There are several aspects of the deep learning models that may have contributed to the improved AUC over the previous machine learning approach to constructing affect detectors for this dataset. In previous detectors, four separate models were built, trained, and evaluated independently while the deep learning model allows all four affective states to be evaluated and updated together with each training sample; such a process likely helps the model determine aspects of the data that help to make more accurate distinctions between each affective state in a temporal sense. Another aspect is in the flexibility of fit supplied by the neural network, allowing the model to capture the high complexity in student affect. This flexibility, however, also exhibits a drawback in terms of lacking interpretability; the large number of parameters and complexity of each model used in this work make it infeasible to study and understand how the model makes its predictions from the features it has available, particularly as it learns from previous time steps. At best, we can understand that the model is relatively better at predicting the more common categories (boredom and concentration) than the more scarce classes (frustration and confusion).

It is desirable to achieve excellent predictive accuracy for the more scarce, yet very important, affective states, in addition to the more common labels. It is possible that a different resampling approach could be more productive, although any resampling approach will be limited by the inter-connection of the observations, leading to non-uniformity across the labels; it is likely that in duplicating sequences containing the scarce labels numerous times, the model overfit to these sequences, which led to poorer extrapolation to unseen data. A possible alternate approach for the iterative refinement of these models would be to send field coders to classrooms working through material that is known to be more confusing and frustrating (e.g. [34]).

One further aspect not addressed by this work is differences introduced by student geographical factors. Earlier affect detectors in ASSISTments were found to perform relatively poorly on rural students when trained on urban and suburban populations [14]. Analyzing how robust deep learning models of affect are to population differences will help us to understand the degree to which these models generalize.

Acknowledgments

We thank multiple current NSF grants (IIS-1636782, ACI-1440753, DRL-1252297, DRL-1109483, DRL-1316736, DGE-1535428 & DRL-1031398) , the US Dept. of Ed (IES R305A120125 & R305C100024 and GAANN), and the ONR.

References

1. Pane, J.F., Griffin, B.A., McCaffrey, D.F., Karam, R.: Effectiveness of Cognitive Tutor Algebra I at Scale. Educational Evaluation and Policy Analysis, 0162373713507480. (2013)
2. Roschelle, J., Feng, M., Murphy, R.F., Mason, C.A.: Online Mathematics Homework Increases Student Achievement. AERA Open, 2(4), 2332858416673968. (2016)
3. Lehman, B., Matthews, M., D'Mello, S., Person, N.: What Are You Feeling? Investigating Student Affective States During Expert Human Tutoring Sessions. In International Conference on Intelligent Tutoring Systems. Springer, 5059. (2008)
4. Craig, S.D., Graesser, A., Sullins, J., Gholson, B.: Affect and learning: an exploratory look into the role of affect in learning with AutoTutor. Journal of Educational Media, 29(3), 241250. (2004)

5. Pardos, Z.A., Baker, R.S., San Pedro, M.O., Gowda, S.M.: Affective States and State Tests: Investigating How Affect and Engagement during the School Year Predict End-of-Year Learning Outcomes. Journal of Learning Analytics 1, 1, 107128. (2014)
6. Pedro, M.O., Baker, R., Bowers, A., Heffernan, N.: Predicting College Enrollment from Student Interaction with an Intelligent Tutoring System in Middle School. In: Proceedings of the 6th International Conference on Educational Data Mining. (2013)
7. D'Mello, S., Lehman, B., Sullins, J., Daigle, R., Combs, R., Vogt, K., Perkins, L., Graesser, A.: A Time for Emoting: When Affect-Sensitivity is and isn't Effective at Promoting Deep Learning. In: International Conference on Intelligent Tutoring Systems. Springer, 245254. (2010)
8. Arroyo, I., Cooper, D.G., Burleson, W., Woolf, B.P., Muldner, K., Christopherson, R.: Emotion Sensors Go To School. In: AIED, 200, pp. 17-24. (2009)
9. Paquette, L., Rowe, J., Baker, R., Mott, B., Lester, J., DeFalco, J., Brawner, K., Sottilare, R., Georgoulas, V.: Sensor-Free or Sensor-Full: A Comparison of Data Modalities in Multi-Channel Affect Detection. International Educational Data Mining Society. (2016)
10. Salmeron-Majadas, S., Santos, O. C., Boticario, J. G.: An Evaluation of Mouse and Keyboard Interaction Indicators Towards Non-Intrusive and Low Cost Affective Modeling in an Educational Context. Procedia Computer Science, 35, 691-700. (2014)
11. Baker, R.S.J.d., Gowda, S.M., Wixon, M., Kalka, J., Wagner, A.Z., Salvi, A., Aleven, V., Kusbit, G., Ocumpaugh, J., Rossi, L.: Towards Sensor-free Affect Detection in Cognitive Tutor Algebra. In: Proceedings of the 5th International Conference on Educational Data Mining, 126-133. (2012)
12. D'Mello, S., Craig, S.D., Witherspoon, A., Mcdaniel, B., Graesser, A.: Automatic Detection of Learner's Affect from Conversational Cues. In: User Modeling and User-Adapted Interaction, 18(1-2), 45-80. (2008)
13. Sabourin, J., Mott, B., Lester, J.C.: Modeling Learner Affect with Theoretically Grounded Dynamic Bayesian Networks. In Proceedings of International Conference on Affective Computing and Intelligent Interaction, 286-295. Springer Berlin Heidelberg (2011)
14. 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), 487501. (2014)
15. 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 Fifth International Conference on Learning Analytics And Knowledge. ACM, 3135. (2015)
16. Williams, R.J., Zipser, D.: A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. Neural Computation 1, 2, 270280. (1989)
17. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural Computation 9(8), 17351780. (1997)
18. Cho, K., Van Merrinboer, B., Bahdanau, D., Bengio, Y.: On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In: Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation. (2014)
19. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. arXiv preprint arXiv:1412.3555. (2014)

20. Socher, R., Lin, C.C., Manning, C., Ng, A.Y.: Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In Proceedings of the 28th international conference on machine learning (ICML-11). 129136. (2011)
21. Min, W., Vail, A. K., Frankosky, M. H., Wiggins, J. B., Boyer, K. E., Wiebe, E. N. et al.: Predicting Dialogue Acts for Intelligent Virtual Agents with Multimodal Student Interaction Data. In 9th International Conference on Educational Data Mining. (2016)
22. Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L.J., Sohl-Dickstein, J.: Deep Knowledge Tracing. In Advances in Neural Information Processing Systems. 505513. (2015)
23. Xiong, X., Zhao, S., Van Inwegen, E.G., Beck, J.E.: Going Deeper with Deep Knowledge Tracing. In 9th International Conference on Educational Data Mining. 545550. (2016)
24. Khajah, M., Lindsey, R.V., Mozer, M.C.: How Deep is Knowledge Tracing? In: Proceedings of the 9th International Conference on Educational Data Mining. (2016)
25. 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), 470497. (2014)
26. Ocumpaugh, J., Baker, R., Rodrigo, M.M.T.: Baker Rodrigo Ocumpaugh Monitoring Protocol (BROMP) 2.0 Technical and Training Manual. Technical Report. Technical Report. New York, NY: Teachers College, Columbia University. Manila, Philippines: Ateneo Laboratory for the Learning Sciences. (2015)
27. Craig, S.D., D'Mello, S., Witherspoon, A., Graesser, A.: Emote Aloud During Learning with AutoTutor: Applying the Facial Action Coding System to CognitiveAffective States During Learning. Cognition and Emotion, 22(5), 777-788. (2008)
28. Theano Development Team.: Theano: A Python framework for fast computation of mathematical expressions. http://arxiv.org/abs/1605.02688 (2016)
29. Dieleman, S., Schlüter, J., Raffel, C., Olson, E. et al.: Lasagne: First release. DOI: http://dx.doi.org/10.5281/zenodo.27878 (2015)
30. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research 15, 1, 19291958. (2014)
31. Estabrooks, A., Jo, T., Japkowicz, N.: A Multiple Resampling Method for Learning from Imbalanced Data Sets. Computational intelligence 20(1), 1836. (2004)
32. Duchi, J., Hazan, E., Singer, Y.: Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research 12, Jul, 21212159. (2011)
33. 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)
34. Slater, S., Ocumpaugh, J., Baker, R., Scupelli, P., Inventado, P.S., Heffernan, N.: Semantic Features of Math Problems: Relationships to Student Learning and Engagement. Proceedings of the 9th International Conference on Educational Data Mining., 223-230. (2016)