Diagrammatic Student Models: Modeling Student Drawing Performance with Deep Learning

Andy Smith, Wookhee Min, Bradford W. Mott, James C. Lester

Center for Educational Informatics
North Carolina State University, Raleigh, NC 27695
{pmsmith4, wmin, bwmott, lester}@ncsu.edu

Abstract. Recent years have seen a growing interest in the role that student drawing can play in learning. Because drawing has been shown to contribute to students' learning and increase their engagement, developing student models to dynamically support drawing holds significant promise. To this end, we introduce *diagrammatic student models*, which reason about students' drawing trajectories to generate a series of predictions about their conceptual knowledge based on their evolving sketches. The diagrammatic student modeling framework utilizes deep learning, a family of machine learning methods based on a deep neural network architecture, to reason about sequences of student drawing actions encoded with temporal and topological features. An evaluation of the deep-learning-based diagrammatic student models suggests that it can predict student performance more accurately and earlier than competitive baseline approaches.

Keywords: Student modeling, intelligent tutoring systems, deep learning

1 Introduction

Diagrams and drawing are fundamental to science learning and understanding. From primary through post-secondary education, students use drawings and graphical representations to make sense of complex systems and as a tool to organize and communicate their ideas to others. Studies have shown that learning strategies centered on learner-generated drawings can produce effective learning outcomes, such as improving science text comprehension and student affect, facilitating the writing process, and improving the acquisition of content knowledge [1] .

Unlike the well studied areas of how people learn from writing text, viewing graphics, and reading, relatively little is known about how generating drawings affects learning. The benefits of drawing arise from actively involving learners in the cognitive processes of selecting, organizing, and integrating the information they have been presented**,** while also requiring both essential and generative processing to mentally connect multiple representations [2]. The benefits of learner-generated drawing are best realized through supports that constrain and structure the drawing activity [3]**.** The act of generating a visual representation can be a cognitively demanding task and, as such, requires scaffolds to guard against excessive and extraneous cognitive load [4]. Examples of effective scaffolds for drawing include providing symbolic drawing elements, guided questioning, and targeted drawing prompts [5].

Intelligent tutoring systems (ITSs) offer a promising approach to addressing the complexity of scaffolding and assessing students' drawing activities. A key feature of many ITSs is the ability to leverage student models that assess knowledge and skills from observed learning activities and support learning based on the predicted competency level in real-time. In this paper, we introduce *diagrammatic student models* that are designed to reason about students' drawing trajectories to generate a series of predictions about their conceptual knowledge based on their evolving sketches. The diagrammatic student modeling framework utilizes deep learning, a family of machine learning methods based on a deep neural network architecture, to reason about sequences of student drawing actions encoded with temporal and topological features. As students draw, an effective diagrammatic student model should be able to not only predict student knowledge but also to weight the contributions of individual drawing actions toward the goal of the sketching activity, as well as monitor student drawing time and efficiency.

To explore diagrammatic student modeling, we have developed a diagrammatic student model for a tablet-based learning environment designed for elementary school science education. In an evaluation, the deep-learning-based diagrammatic student model was compared against multiple baseline models using an annotated corpus of elementary student science drawings. The evaluation shows that the deep-learning-based diagrammatic student model predicts student drawing performance more accurately than baseline models, as well as requires fewer actions to converge on the correct prediction.

This paper is structured as follows. Section 2 discusses related work on analyzing student drawing and predicting student performance. Section 3 describes the tablet-based learning environment that was used to collect the drawing dataset of symbolic sketches from elementary students. Section 4 presents the diagrammatic student modeling framework used for the predictive modeling task as well as the stacked autoencoder pre-training technique used in deep learning. Section 5 describes an evaluation of the system compared to other predictive models.

2 Related Work

Research on student modeling has explored a variety of computational frameworks. Several families of models perform a running estimate of students' skills and knowledge based on their previous performance across multiple problems, including techniques such as Bayesian knowledge tracing [6] and performance factor analysis [7]. Sabourin et al. combined differential sequence mining with a dynamic Bayesian network to perform early prediction of students' self-regulated learning behaviors in an educational game [8]. Chi et al. used reinforcement learning to identify features from student-tutor interactions and induce pedagogical strategies [9]. Other systems have explored features based on analysis of trace logs to predict transfer of inquiry skill [10], predict gaming the system behaviors [11], and predict user goals [12].

Similar to the techniques used in this work, a deep learning technique leveraging denoising autoencoders has been used to accurately identify intermediate player goals in an open-ended science mystery game [13]. The approach presented here utilizes deep learning techniques on the problem of modeling students drawing activities.

Outside of the student modeling community, several efforts have made significant progress on machine understanding of student drawing artifacts. Mechanix uses free-hand sketch recognition capabilities to convert student statics drawings into free-body equations that the system can then compare to a target solution and provide basic forms of feedback [14]. Van Joolingen et al.'s SimSketch system merges free-hand sketching with modeling and simulation of science phenomena by segmenting the drawing into distinct objects that are then annotated by the user with a variety of behaviors, attributes, and labels [15]. Students can then run a simulation based on their drawing and see the results before revising their sketch. SimSketch has been evaluated in a planetarium setting and been shown to be both a useable and engaging system for visitors. Researchers have also investigated CogSketch, an open-domain sketch understanding engine, to compare the drawings of expert and novice users to analyze differences in final drawings, as well as differences in the way the drawings are created [16].

3 The LEONARDO Science Notebook

Recent years have seen a growing interest in introducing science notebooks into elementary science classrooms [17]. Science notebooks capture students' inquiry-based activities in both written and graphical form, potentially providing a valuable source of both diagnostic and prognostic information. However, because elementary school teachers sometimes have limited training in science pedagogy, they often struggle with effectively using science notebooks in classroom learning activities.

For the past four years our laboratory has been developing a digital science notebook for elementary school students, LEONARDO (Figure 1), which runs on tablet computing platforms. LEONARDO integrates intelligent tutoring systems technologies into a digital science notebook that enables students to graphically model science phenomena with a focus on the physical and earth sciences. LEONARDO features a pedagogical agent that supports students' learning with real-time problem-solving advice. LEONARDO is designed for use in the classroom in conjunction with popular science kits, and it is aligned with the Next Generation Science Standards for elementary school science education. It has been used in schools in more than ten states in the US.

Throughout the inquiry process, students using LEONARDO are invited to create symbolic sketches of different types, including electrical circuits. Given the challenges of machine recognition for freehand sketch, as well as concerns of excessive cognitive demand for fourth graders working in such an unstructured space [18], LEONARDO supports symbolic drawing tasks. To preserve the generative processing thought to be of great benefit for learner-generated drawings strategies, each activity begins with a blank workspace so that the representations must be created from scratch. Students then choose from a variety of semantically grounded objects and place them at various points on the drawing canvas. For example, objects for the electricity unit include light bulbs, motors, switches, and batteries. Students can then place wires on the drawing canvas and connect the various objects that together simulate proper electrical behavior. This approach enables students to focus on choosing the appropriate circuit elements and creating the appropriate circuit topology rather than having to concentrate on free-hand sketching complex objects such as motors and switches. Drawing tasks vary in complexity from replicating a picture of a circuit held up by the pedagogical agent, to recreating a circuit made during a physical investigation, to creating more complex circuits designed to increase their understanding of series and parallel circuits.

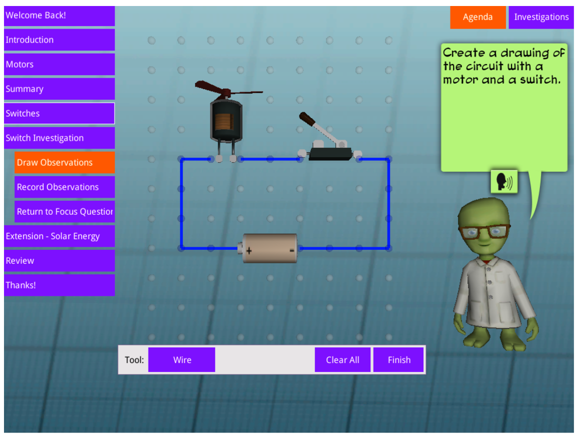

Figure 1. LEONARDO digital science notebook

4 Diagrammatic Student Modeling

To scaffold and assess drawing activities such as those supported by the LEONARDO digital science notebooks, a *diagrammatic student model* could reason about students' drawing progress and infer students' conceptual knowledge. We introduce diagrammatic student models by first discussing topology-based methods they use to analyze and compare student drawings. We then describe how they can predict student conceptual knowledge based on their drawing trajectories.

To analyze student drawings the diagrammatic student modeling system first translates them into a more abstract representation (Figure 2) [19]. It takes as input trace logs from students' work and extracts student actions at a level of granularity capable of producing replay-quality representations of the activities. From these actions it can reconstruct the state of the student drawing at each point in the activity. After each student action, a topological representation of the drawing is constructed using the set of objects and locations combined with a simulation engine that supports the querying of topological features of the drawing.

These topological features are used to generate a labeled graph representation of the drawing. The first step in the translation from drawings to topological graphs is encoding of the primary elements for the domain. For the domain of circuits, we define this as non-wire circuit elements. Circuit elements are represented as nodes in the graph. Because there are only two points where each node can interact with other objects in the drawing space, each node is connected to two child nodes representing its contact points. Nodes are then labeled by type, for circuits the supported element types are Light bulb, Motor, Switch, or Battery.

After creating the nodes of the graph, edges are generated based on relationships between elements. For this domain the only relationship encoded is electrical connectivity, though for other domains additional relationships could be encoded such as 2D spatial relations such as near/far, overlapping, or containing. For each contact point in the graph, the simulation engine uses a depth first search with resistance as the cost function to return all other contact points reachable with a zero resistance path. If one or more paths exist between contact points, they are then connected with a single edge in the graph.

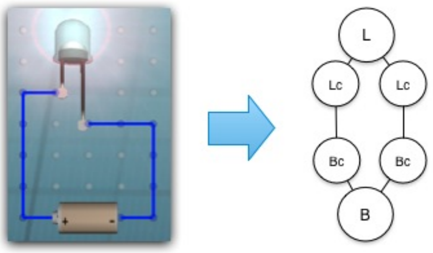

Figure 2. Circuit encoded as topological graph

Topologies can then be compared using a modified form of edit distance, which determines the number of operations needed to transition from one graph to a target graph. Edit distance measures the number of element additions, element deletions, edge additions, and edge deletions needed to match two topologies. While traditional string edit distances tend to also utilize substitution, we chose to treat this instead as deleting an element, then adding a new one because this is the path students would take to modify their drawing. While edit distance allows an ITS to compare individual drawings, it is also critical for the system to be able to incorporate information about how the student arrived at that point. For example, consider a student with a blank workspace. By assessing only the current state of the drawing, the system would have no idea whether the student is yet to start drawing, or has been attempting unsuccessfully to complete the circuit for several minutes.

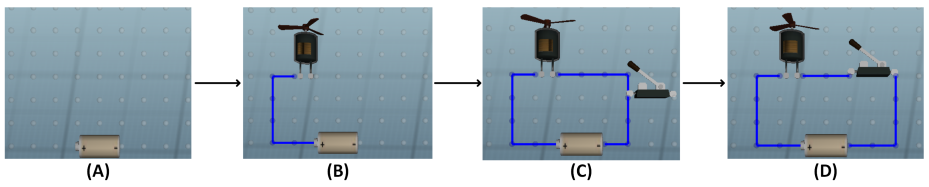

Figure 3. Sample drawing progression

An important capability of diagrammatic student models is accurately predicting the level of performance a student will attain in a drawing task. For example, using the drawing progression shown in Figure 3, the diagrammatic student modeling system seeks to predict the quality of the final submitted drawing (D), based on each drawing action (e.g., add, remove, rotate, drag). We treat this as a multiclass classification problem, in which the model seeks to predict the most likely student outcome state given a sequence of drawing actions. Outcome states for the diagrammatic student modeling system are defined using a clustering approach previously shown to align with human classification [19]. Students' final drawings are grouped into 3 clusters: Correct, Near-Miss, and Far-Miss. For each drawing action by a student, we aim to predict which of these clusters the final drawing will belong to. The input to the classification task is encoded as a feature vector constructed from the student drawing after every action. The feature vectors consist of 6 features:

- **Time:** Number of seconds from the start of the activity.
- **Action Step:** Number of actions performed by the student thus far.
- **Extra Elements:** Number of superfluous non-wire elements on the workspace.
- **Missing Elements:** Number of required elements missing from the workspace.
- **Extra Topological Connections:** Number of superfluous connections between non-wire elements.
- **Missing Topological Connections:** Number of missing connections between non-wire elements.

The output the diagrammatic student model is a prediction of which cluster the action sequence will culminate in. It is intuitive that predicting student outcomes could benefit from observations based on the sequence of actions preceding the current action. While this information is partly encoded in the drawing features, we further investigated it by augmenting the input feature vector with *n*-gram encodings. For *n*-gram encodings, the feature vector for a given action was combined with the feature vectors of the previous *n*-1 actions. For example, the input feature for a 1-gram encoding would contain 6 elements, the input vector for a 5-gram model would include 30 elements, and a 10-gram model would include 60 elements in total.

For our classifier we leverage deep learning. *Stacked autoencoders* (SAEs), a pre-training technique, are investigated to avoid issues of underfitting that are often encountered when training deep neural networks. The following sections provide background on deep learning in general as well as the specific techniques used in our model.

4.1 Deep Learning Overview

Deep learning (DL) is a family of machine learning techniques that seek to learn multiple levels of higher-level features from lower-level data (e.g., pixels in image classification, acoustic inputs in speech recognition) through deep neural networks. A key advantage of DL is its feature extraction capabilities, which reduces the need for feature engineering by human experts that is often expensive in terms of time and effort. The majority of deep learning techniques and model structures are based on artificial neural networks (ANNs). ANNs' structures are typically based on a collection of nodes, often referred to as neurons, and weighted edges propagating values to neurons in the next layer using both linear and non-linear transformations. ANNs have multi-layered structures that consist of an input layer, one or more hidden layers, and an output layer. As a method of training ANNs, a variety of backpropagation methods have been examined for model optimization, including stochastic gradient descent and batch gradient descent methods. Unfortunately, these algorithms were often not scalable to deep networks with multiple hidden layers. The resulting deep networks often performed worse than single hidden layer networks due to their increased vulnerability to converging on poor local optima during training [20]. DL systems have successfully set state-of-the-art benchmarks in a variety of tasks in fields such as computer vision [21], and sentiment analysis of text [22].

The diagrammatic student models utilize the SAE technique. SAEs operate based on a set of *autoencoders* (AEs) that are characterized by encoding and decoding steps using an unsupervised criterion [20]. The encoding step takes an input vector and projects it onto an output space using an activation function. A decoder step performs the opposite action, in which it decodes the projected output back to the original input with the objective of minimizing the reconstruction error of the input data. By situating an "encoder" and "decoder" in a three-layer network and optimizing the network using backpropagation, one can obtain a set of initialized weights that can be used for the connections between the first two layers. Once a weight configuration is initialized between the first two layers, the hidden layer activations based on the previous AE will be used as the input layer in the next AE training as a part of the larger network. This iterative method of initializing weights between two layers and combining pre-trained weights across all layers is referred to as *stacked autoencoders*. Once the weights for all hidden layers have been pre-trained, the entire network can be optimized using standard backpropagation techniques in a supervised fashion to fine-tune the network for the classification task.

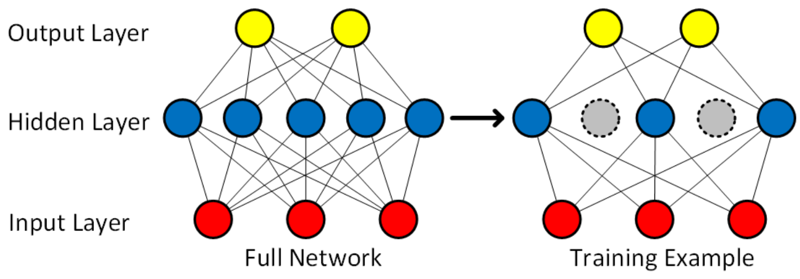

Figure 4. Dropout training

While significantly improving the predictive performance of multi-layer neural networks, SAEs can induce networks that are highly overfit to the training set resulting in high generalization error. One approach to overcoming this problem would be to train a large number of networks using different portions of the training data and averaging their predictions to classify new data. Unfortunately, this requires a large amount of training data, and greatly increases both the training time, and prediction time for a trained network to an impractical level. Researchers led by Geoffrey Hinton developed a method called dropout to mimic this behavior, while only requiring the training of one network [23]. As illustrated in Figure 4, dropout works by probabilistically removing neurons from the network during training. For each training example, a pre-defined proportion (*p*) of neurons are stochastically dropped from each layer, and both the input and output weights connected to these neurons are correspondingly removed from the network. The weights are then learned for the thinned network using standard backpropagation, and the process is repeated for each training batch. Once training a network is completed, all the weights are downscaled by (1-*p*) as an adjustment process. Empirical analyses demonstrate that a network made up of weights combined from a large number of thinned networks effectively lower generalization error when training deep neural networks [23].

4.2 Model Training and Inference

The open-source DeepLearnToolbox [24] was used to create and test the networks for the diagrammatic student models. While some general guidelines exist, the optimal structure and parameters for a multi-layer network must be determined empirically for each data set. We explored the space of models by conducting a grid search across the following parameters:

- N-grams: 1, 5, 10
- Dropout Rate: 0, .25, .5
- Hidden Layers: 2, 3

All other parameters were fixed for all trials. The number of neurons per hidden layer was set to 150, the activation function was set to the sigmoid function, the backpropagation algorithm was set to stochastic gradient descent over 100 epochs with the learning rate set to .2 for all layers in both the pre-training and fine-tuning steps. Each model contains 3 output nodes that represent class labels (Correct, Near-Miss, and Far-Miss) inferred through this predictive model. To compare models, student-level 10-fold cross validation was conducted using 10 train/test pairs created with data from each student appearing in exactly one of the 10 test sets.

5 Evaluation

To evaluate our model, a corpus of fourth grade symbolic drawings was collected with the LEONARDO system running on iPads in elementary classrooms in North Carolina and California in the spring of 2014. Specifically, the student drawings were from an exercise requiring the students to create a circuit involving a switch, motor, and battery connected in series. After data cleaning, drawings from 403 students were used for the analysis. For this exercise, students spent an average of 4 minutes and 24 seconds in the drawing environment, completing an average of 171 actions. The drawings were automatically scored in comparison to normative models after each action, and the final drawings were clustered using the procedure described in [19]. The clustering process grouped student answers into 3 groups: Correct, Near-Miss, and Far-Miss. Far-Miss was the most common outcome, accounting for 40.01% of final drawings, and 37.34% of total drawing actions. 37.57% of students correctly completed the exercise (30.19% of actions), and 22.42% of student answers were labeled as Near-Miss (32.47% of actions). For each of the actions in the drawing environment, the model attempted to predict the cluster of the final drawing.

The deep-learning-based (DL) diagrammatic student model, which used SAE pre-training, was evaluated against other diagrammatic student models that used a J48 Decision Tree, a Naïve Bayesian classifier, and a Support Vector Machine (SVM) with a Gaussian kernel as well as a most common class baseline. In total 69,979 actions were classified by each model. Each model was trained on a single action based inputs (1-gram encoding), and a sequence of actions based inputs (5 and 10-gram encoding). Because the effectiveness of SVM depends largely on its hyperparameter settings as in deep neural networks, a grid search of C {*.5,1,2,4,8*} and γ {.0625 , .125, .25, .5, 1, 2} was conducted for fair comparisons, and the model with the highest accuracy is reported out of models based on all possible hyperparameter combinations within the ranges [25]. All models were evaluated with the same 10 train/test pairs, with the overall accuracies presented below representing the average across the 10 folds.

Table 1. Model accuracy rates

	Accuracy
Most Common Class	37.34%
Naïve Bayes	45.88%
Decision Tree	45.87%
SVM	50.85%
DL	**55.68%**

Table 1 shows the best performing model for each technique. All models performed best using the 10-gram encoding, showing the importance of including previous student actions in the task of predicting students' knowledge based on their evolving sketches. For the SVM model, a C of 4 and a γ of 2 produced the best model. For the DL model, 2 hidden layers and a dropout rate of .5 produced the best model, with an accuracy of 55.68%. Due to our data not conforming to the normal distribution, we computed significance using two non-parametric tests. Both a Friedman test ($p < .05$) and a McNemar post hoc test ($p < .0001$) elicit that the DL model significantly outperformed the top-performing baseline technique (SVM).

While accuracy provides one metric for evaluating classifiers, it is also important to evaluate how quickly and accurately a model converges on a solution. To achieve this, we used two metrics from the goal recognition literature [26]: convergence rate and convergence point. *Convergence rate* (CR) calculates what percentage of students our model had an accurate last prediction for, and thus higher is better for this metric. *Convergence point* (CP) measures how early in the sequence of actions our model was able to converge on the correct prediction, with a lower percentage indicating earlier convergence. Table 2 shows that the DL model outperformed SVM with respect to both CR and CP, and importantly its significantly lower CP score demonstrates DL's effectiveness as an early predictor of student performance.

Overall, the results show that the deep-learning-based diagrammatic student model more accurately predicted student drawing outcomes compared to several competitive models on the corpus of fourth grade circuit drawings. A possible explanation for the

DL model's outperformance compared to the competing techniques is DL's ability to discover high-level representations processed through hierarchical abstraction of low-level data, an attribute closely related to the latent nature of the student modeling task. Further, the model was also able to converge to the correct prediction sooner in the sequence of actions than the baseline models, showing great potential for informing an intelligent tutoring system of student's outcomes early in the drawing process.

Table 2. Convergence rate and convergence point

	Convergence Rate	Convergence Point
SVM	78.3%	68.5%
DL	78.8%	**55.4%**

6 Conclusions and Future Work

Drawing is critical to science learning, and it provides a rich source of diagnostic and predictive information about a student's understanding. However, it has been shown that without proper support, students can be overwhelmed by the process of drawing and fail to experience its benefits. It appears that if ITSs could scaffold the drawing process and perhaps use it as a source for assessment, they could enable students to experience the benefits that drawing appears to offer. To pave the way toward ITSs that support student drawing, we have introduced diagrammatic student models, which analyze student drawing artifacts and use these analyses to predict drawing performance.

In this paper we have reported on the investigation of a diagrammatic student modeling framework based on deep learning. The deep-learning-based diagrammatic student model utilizes temporal and topological features of students' drawing trajectories to predict students' performance from their drawing actions. In an evaluation in which the deep-learning-based framework was compared with several competing models, the deep-learning approach was most accurate, and outperformed the other models on both convergence rate and convergence point. The strong performance of the model suggests that deep-learning-based diagrammatic student modeling offers significant potential for modeling student drawing activities, which will play an increasingly important role in education with the proliferation of tablet computing devices.

The encouraging results suggest several promising directions for future work. First, it will be important to begin experimenting with embedding diagrammatic student models into ITSs that support drawing in order to better evaluate the practical requirements for model accuracy in improving learning outcomes. Similarly, it will be important to develop techniques that allow diagrammatic student models to predict intermediate sub-goals in drawing activities. These techniques will contribute directly to misconception detection functionalities. Finally, it will be important to explore alternative deep learning architecture to support drawing analyses. For example, recurrent neural networks have proven effective for sequential data and hence may perform well in diagrammatic student modeling considering the highest accuracy was achieved by the 10-gram encoded models.

Acknowledgments. The authors thank our colleagues from the LEONARDO project for their contributions to the LEONARDO project: Eric Wiebe, Courtney Behrle, Mike Carter, Angela Shelton, Samuel Leeman-Munk, and Robert Taylor. This material is based upon work supported by the National Science Foundation under Grant No. DRL-1020229. Any opinions, findings, and conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Van Meter, P., Garner, J.: The Promise and Practice of Learner-Generated Drawing: Literature Review and Synthesis. Educational Psychology Review. 17, 285–325 (2005).
2. Schwamborn, A., Mayer, R.E., Thillmann, H., Leopold, C., Leutner, D.: Drawing as a Generative Activity and Drawing as a Prognostic Activity. Journal of Educational Psychology. 102, 872–879 (2010).
3. Schmeck, A., Mayer, R.E., Opfermann, M., Pfeiffer, V., Leutner, D.: Drawing Pictures During Learning from Scientific Text: Testing the Generative Drawing Effect and the Prognostic Drawing Effect. Contemporary Educational Psychology. 39, 275–286 (2014).
4. Verhoeven, L., Schnotz, W., Paas, F.: Cognitive Load in Interactive Knowledge Construction. Learning and Instruction. 19, 369–375 (2009).
5. Zhang, H., Linn, M.: Using Drawings to Support Learning from Dynamic Visualizations. Proceedings of the 8th International Conference of the Learning Sciences. pp. 161–162. Utrecht, The Netherlands (2008).
6. Corbett, A.T., Anderson, J.R.: Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modelling and User-Adapted Interaction. 4, 253–278 (1994).
7. Gong, Y., Beck, J.E.: Looking Beyond Transfer Models: Finding Other Sources of Power for Student Models. Proceedings of the 19th International Conference on User Modeling, Adaptation and Personalization. pp. 135–146. Girona, Spain (2011).
8. Sabourin, J., Mott, B., Lester, J.: Utilizing Dynamic Bayes Nets to Improve Early Prediction Models of Self-regulated Learning. Proceedings of the 21st International Conference on User Modeling, Adaptation and Personalization. pp. 228–241. Rome, Italy (2013).
9. Chi, M., Vanlehn, K., Litman, D., Jordan, P.: Inducing Effective Pedagogical Strategies using Learning Context Features. Proceedings of the 18th International Conference on User Modeling, Adaptation and Personalization. pp. 147–158. Big Island, HI, USA (2010).
10. Sao Pedro, M.A., De Baker, R.S.J., Gobert, J.D., Montalvo, O., Nakama, A.: Leveraging Machine-learned Detectors of Systematic Inquiry Behavior to Estimate and Predict Transfer of Inquiry Skill. User Modelling and User-Adapted Interaction. 23, 1–39 (2013).

11. Muldner, K., Burleson, W., Van De Sande, B., Vanlehn, K.: An Analysis of Students' Gaming Behaviors in an Intelligent Tutoring System: Predictors and Impacts. User Modelling and User-Adapted Interaction. 21, 99–135 (2011).
12. Armentano, M.G., Amandi, A.A.: Modeling Sequences of User Actions for Statistical Goal Recognition. User Modelling and User-Adapted Interaction. 22, 281–311 (2012).
13. Min, W., Ha, E.Y., Rowe, J.P., Mott, B.W., Lester, J.C.: Deep Learning-Based Goal Recognition in Open-Ended Digital Games. Tenth Artificial Intelligence and Interactive Digital Entertainment Conference. pp. 37–43. Raleigh, NC (2014).
14. Valentine, S., Vides, F., Lucchese, G., Turner, D., Kim, H., Li, W., Linsey, J., Hammond, T.: Mechanix: A Sketch-Based Tutoring System for Statics Courses. Proceedings of the Twenty-Fourth Conference on Innovative Applications of Artificial Intelligence. Toronto, Ontario, Canada (2012).
15. Joolingen, W. van, Bollen, L., Leenaars, F.: Using drawings in knowledge modeling and simulation for science teaching. Advances in Intelligent Tutoring Systems. pp. 249–264 (2010).
16. Jee, B., Gentner, D.: Drawing on experience: Use of sketching to evaluate knowledge of spatial scientific concepts. Proceedings of the 31st Annual Conference of the Cognitive Science Society. Amsterdam, The Netherlands (2009).
17. Ruiz-Primo, M. a., Li, M., Ayala, C., Shavelson, R.J.: Evaluating Students' Science Notebooks as an Assessment Tool. International Journal of Science Education. 26, 1477–1506 (2004).
18. Wiebe, E.N., Madden, L.P., Bedward, J.C., Carter, M.: Examining Science Inquiry Practices in the Elementary Classroom through Science Notebooks. Presented at NARST Annual Meeting. Garden Grove, CA (2009).
19. Smith, A., Wiebe, E., Mott, B., Lester, J.: SketchMiner : Mining Learner-Generated Science Drawings with Topological Abstraction. Proceedings of the Seventh International Conference on Educational Data Mining. pp. 288–291. London, U.K. (2014).
20. Bengio, Y., Lamblin, P.: Greedy Layer-wise Training of Deep Networks. Advances in neural information processing systems. 19, 153 (2007).
21. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems. 1097–1105 (2012).
22. Socher, R., Pennington, J., Huang, E.: Semi-supervised recursive autoencoders for predicting sentiment distributions. In: Linguistics, A. for C. (ed.) Proceedings of the Conference on Empirical Methods in Natural Language Processing. pp. 151–161 (2011).
23. Srivastava, N., Hinton, G.: Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research. 15, 1929–1958 (2014).
24. Palm, R.: Prediction as a candidate for learning deep hierarchical models of data. Technical University of Denmark. (2012).
25. Hsu, C., Chang, C., Lin, C.: A Practical Guide to Support Vector Classification. 1, 1–16 (2003).
26. Blaylock, N., Allen, J.: Hierarchical Goal Recognition. Plan, Activity, and Intent Recognition Theory and Practice. pp. 3–31 (2014).