Unsupervised Deep Autoencoders for Feature Extraction with Educational Data

Nigel Bosch
University of Illinois at Urbana-Champaign
1205 West Clark Street
Urbana, IL, 61801, USA
pnb@illinois.edu

Luc Paquette
University of Illinois at Urbana-Champaign
1310 South 6th Street
Champaign, IL 61820, USA
lpaq@illinois.edu

ABSTRACT

The goal of this paper is to describe methods for automatically extracting features for student modeling from educational data, and students' interaction-log data in particular, by training deep neural networks with unsupervised training. Several different types of autoencoder networks and structures are discussed, including deep neural networks, recurrent neural networks, variational autoencoders, convolutional neural networks, and asymmetric network structures. Autoencoder networks are trained to find low-dimensional, predictive embeddings of raw interaction-log data. These embeddings are then entered into a model as features for supervised classification tasks. We discuss the implications for training these network structures with educational data, including peculiarities that arise for interaction-log data that are not as commonly encountered in domains such as computer vision and natural language processing. Methods for evaluating the network training process are also discussed, with examples showing the importance and efficacy of visualizing neuron activations to diagnose common problems encountered during training and verifying that embedded representations of data follow desired distributions. We provide an example of how automatically extracted features can be used in a classification problem for the detection of student affect. In this example, student boredom was detected at levels above chance (area under the receiver operating characteristic curve = .673 versus .5 chance). Finally, opportunities for future work are discussed, including transfer learning and semi-supervised methods.

Keywords

Unsupervised learning, deep learning, autoencoder, affect detection, student modeling, student interactions

1. INTRODUCTION

Modeling students in computer-based education platforms is important for several reasons. A wide variety of educational strategies and assessment capabilities are made possible by automatic detection of students' affective, cognitive, and behavioral states—such as when a student is bored, engaged, off-task, or understanding a particular concept [8]. For example, software can automatically target interventions when students stop paying attention [10], track cognitive workload within a lesson [27], predict when a student is going to drop out of a course [39], and many more possibilities.

Toward these goals, researchers have explored machine learning methods for student modeling based on data such as interaction logs [1, 32], facial features [4, 38], physiology [3, 13], and other modalities (see [9, 42] for reviews). Various machine-learned models such as Bayesian models [16], support vector machines [25], and logistic regression [41] have been employed. Recently, deep learning methods (neural networks with multiple non-linear hidden layers in their structure) have shown promise for student modeling applications [33], and are growing ever more popular in the broader machine learning community [17]. Applying deep learning methods to educational data is promising for the potential to not only improve accuracy but enable new possibilities for student modeling similar to the successes seen in other domains.

Some of these possibilities include automatic feature extraction from raw data, learning the structure of time series and other sequential data, and more. Deep learning also offers opportunities for transfer learning across related domains. For example, neural networks trained to detect objects in images can be adapted to recognize students' facial expressions [29]. In this manner, such models utilize much of the information learned from recognizing objects to avoid the need for millions of instances of labeled facial expression data, which are typically difficult to acquire.

Deep learning approaches thus have much to offer for advancing the state of the art in student modeling for educational software. However, deep learning methods are still nascent in the education domain, and there are some unique challenges that will have to be addressed.

One of the biggest challenges of training deep neural networks is that they are very "data hungry", typically requiring a large number of labeled training instances to fit the network parameters. However, when modeling student features such as test results, course grades, graduation outcomes, and other outcomes it is impossible to obtain more than a few data points per student. For example, Klingler et al. [21] classified students as having developmental dyscalculia or not, which, by definition, results in one label per student. Similarly, when modeling student affect, cognition, or behavior manual labeling of students is typically necessary [24, 31, 38]. In these situations labeled instances are also limited to hundreds or thousands of instances rather than the hundreds of thousands or millions that are frequently employed for training supervised deep networks [17].

Additionally, the most common applications of deep learning (e.g., computer vision, natural language processing) deal with homogenous data. For example, when recognizing objects in an image, the inputs to a neural network are the numeric values of the pixels in the image. Each pixel is the same type of data as all the other pixels in the image (e.g., a number from 0-255). Each pixel is also strongly related to other pixels in its local neighborhood. These properties allow for quick development of neural networks that are large but made up of many identical connections, with minimal data pre-processing required. On the other hand, some types of educational data, such as logged interaction behaviors, do not share these properties. An interaction log might be made of up columns describing timing information in milliseconds, integer counts of different actions taken, and proportions of items completed. The varied data types make transfer learning across domains all but impossible, and different scales necessitate preprocessing to prevent large-valued inputs from dominating the network.

In this paper, we detail strategies for addressing the issue of sparse labels in temporally fine-grained educational data via a combination of supervised and unsupervised deep learning methods. We also propose methods for dealing with the unique difficulties of training such models with educational data, and interaction-log data in particular.

This paper is novel in several respects. We offer an overview of how to construct several different types of unsupervised neural network models for educational interaction data for the first time, including fully connected, recurrent, variational, and convolutional autoencoder networks. We also introduce a method for improving unsupervised representations of interaction data by predicting future sequences. The code for training and testing models described in this paper is available online (https://github.com/pnb/dlwed17).

2. RELATED WORK

This section includes brief discussions of research related to deep learning in education and autoencoders for deep feature extraction. More specialized methods for feature extraction are detailed in the methods section.

2.1 Deep learning in education

Although deep learning research is not yet well explored for educational purposes, there have been a few studies that employed it for student modeling. For example, Metallinou et al. [26] trained a deep neural network model for speech recognition to improve assessment of children learning English. They measured speech recognition accuracy with word error rate (WER), where 0% implies perfect speech recognition (no errors). Their deep neural network model achieved 19.3% WER versus 27.8% WER for a more traditional Gaussian mixture model. This research demonstrates a fairly straightforward implementation of deep learning methodologies for speech recognition [7] adapted to an educational environment.

Piech et al. [33] employed deep learning to improve student knowledge tracing compared to traditional methods (Bayesian knowledge tracing [6]). They utilized logged sequences of a problem identifier and problem correctness as inputs to predict the probability of students successfully answering a problem. Area under the receiver operating characteristic curve (AUC) served to measure prediction accuracy. Chance-level AUC is .5, while .0 represents completely incorrect classification and 1.0 represents perfect classification. The deep knowledge tracing method resulted in an impressive AUC = .85 on one dataset, compared to the traditional method on the same data resulting in AUC = .68. A similar improvement in accuracy was observed on another dataset, where the deep learning method resulted in AUC = .86 versus .69 for the traditional method. However, later research motivated by deep knowledge tracing showed that adding additional attributes (designed to model unaccounted-for aspects of student knowledge) to the traditional Bayesian method improved accuracy to be equivalent to the deep learning method [19]. This illustrates one of the advantages of the deep learning approach as well, in that attributes could be automatically discovered without explicitly modeling them (e.g., similarity between exercises and differences between students).

Tang et al. [37] constructed deep neural networks to generate essay text with a neural network trained on students' essays. They also predicted future student actions in a sequence from their past actions. These results were not better than simply predicting the majority class student action, but demonstrates a first effort toward integrating student interaction data into a deep learning framework. Additionally, the method of predicting future student actions from past actions is conceptually similar to autoencoding.

2.2 Autoencoding neural networks

Autoencoders are a type of neural network in which the input to the network is the same as the output. The network structure can be designed so that the center of the network is smaller (represented by fewer numbers) than the input. An autoencoder thus learns a compact representation of the input, with no need for labels. It is thus an unsupervised learning method that can be applied to vast amounts of raw data without intractable annotation processes. In a seminal study on autoencoders, Hinton et al. [11] showed that autoencoders produced compact representations of images that are more accurate representations for reconstructing the original images than principal components analysis [40], a statistical method frequently employed to reduce data dimensionality.

One of the biggest advantages of autoencoders is that they can be adapted to learn different types of patterns based on the domain knowledge of experts. Speech recognition research offers an example of such adaptation. Maas et al. [23] developed a recurrent network architecture, in which neurons are connected through time, to model the temporal dependencies that are inherently present in audio (i.e. every audio sample is highly related to the one before and after it). They then trained autoencoders to remove noise from audio by training a network with excess noise added to the input (but not the output), so that the network learned to create noise-free audio from noisy audio. They found that adding a recurrent structure to the deep neural network dramatically improved the accuracy of noise-removing autoencoders compared to non-recurrent network structures (mean squared error decreased from 47.2% to 30.7% on average across all tasks).

2.3 Current paper

Related work involving deep learning in education is limited. The current paper describes applications of related work to educational interaction-log data. Specifically, we discuss applications of different neural network architectures when designing deep autoencoders for extracting compact representations of student interactions. We then show the efficacy of one such representation for detecting student boredom in a simple case study.

3. AUTOENCODER NETWORK STRUCTURES

The unsupervised deep autoencoder methods that we focus on in this paper are primarily intended to extract embedded representations (embeddings) that can then serve as features for supervised classification.

3.1 Deep neural networks (DNNs)

In its simplest form, a deep neural network simply has an input layer, multiple hidden layers, and an output layer. The connections between neurons in different layers in a simple network are typically fully connected, i.e. there are connections between every pair of neurons in consecutive layers. This structure has no built-in representation of temporal or spatial locality, as every input is treated equally and connected with every other neuron with no regard for which connections are likely to be meaningful. This strategy works well for simple problems, but educational data typically has temporal relationships that can be exploited to extract more effective embeddings of interaction data. The other network structures discussed below are also deep (having multiple hidden layers between input and output layers) but have different connections and neurons that are designed to exploit specific structures in data.

3.2 Recurrent neural networks (RNNs)

RNNs are a class of network structure in which the neurons at each layer are not only connected to neurons from adjacent layers, but also receive the inputs from the same layer at the previous step in a sequence (e.g., the moment before in a time series). Figure 1 illustrates the connections between neurons across steps in sequential data. With these connections the network is capable of learning patterns that develop over time or some other dimension. In the case of interaction-log data, RNNs can be utilized to learn sequences that develop over time. For example, an RNN autoencoder can learn to encode patterns of a student's correct/incorrect responses to questions, or to recognize sequences of repeated hint usage in a learning environment.

One issue commonly encountered when training RNNs with backpropagation is that the error term used to adjust weights has a tendency to trend toward zero or infinity quickly due to modification from activation functions in the neurons, and thus does not allow training over many steps in sequential data. Long short-term memory (LSTM) neurons solve this problem by storing values in the network and passing them to the next (or previous) step in a sequence using no activation function. The absence of an activation function across steps in the sequence implies that the values are not be modified, and so can be passed on for an arbitrary number of steps, thus allowing the network to learn relationships across longer sequences [12].

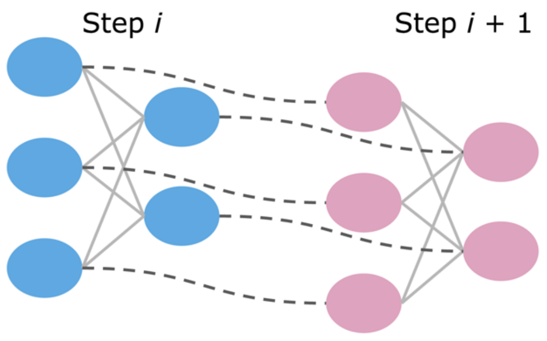

Figure 1. Connections between steps in an RNN

3.3 Convolutional neural networks (CNNs)

RNNs are not the only way to capture patterns in sequences, however. A CNN is a type of neural network that applies a small group of neurons (a filter) across the data for every input sample. The neurons thus learn features that are defined by local patterns (e.g., an eyeball in an image) but which could occur anywhere in the input. Typically, many such filters are applied to the data, and initialized randomly so that they eventually learn to detect different features.

As mentioned previously, educational data are frequently composed of features that are heterogeneous data types and not locally related to other features in the dataset. Thus, applying convolution across features makes little sense. However, we can apply one-dimensional convolution across time, because each step in time series data is semantically the same as the other steps (i.e. has the same inputs). A CNN is thus relevant to learning from students' interaction data by finding patterns of interaction that are notable but whose position within the sequence of actions is less important to capture. This is analogous to a bag of *n*-grams model of natural language processing, where short groups of words in a sentence serve as features without capturing exactly where in the sentence those groups of words were.

3.4 Variational autoencoders (VAEs)

RNNs and CNNs are ways of adapting neural networks to capture temporal patterns in educational data, but there are other considerations as well. Autoencoders are commonly trained to minimize the difference between reconstructed inputs and the actual inputs (i.e. reconstruction loss). This may result in highly compact, accurate embeddings that represent the raw data well, but embeddings can have unpredictable distributions that may not be suitable for feeding into supervised classifiers. VAEs address this potential issue by not only minimizing the reconstruction loss, but also minimizing the difference between distributions of values in the embeddings compared to a Gaussian distribution (typically with zero mean and unit variance) [20]. The network thus learns to compactly represent student data using features that are normally distributed, which could theoretically improve the accuracy of supervised classification for some classifiers (e.g., Gaussian naïve Bayes) that assume data are normally distributed.

A further possible application of VAEs is for generating new data. Since the embeddings are distributed predictably, it is possible to synthesize student actions (e.g., student interactions in a system) by sampling values from a normal distribution and feeding them into the embedding layer in the neural network. We do not explore this possibility further in the current paper, but it might be of benefit for future research on student simulation.

Training VAEs can be difficult because of the two-part optimization problem. If the distribution component is weighted too highly, the network simply learns to predict Gaussian values with no regard for reconstructing the input data correctly. A common solution to this issue is to "warm up" the optimization function by initially optimizing only reconstruction of inputs, and then gradually imposing the Gaussian restriction on the embedding layer over the course of several epochs (iterations though all the training data) [34]. This approach is problematic for training autoencoders on educational data with millions of rows of student interactions, however. Such a VAE is typically relatively well-fit by the end of the first epoch, so the network has already learned a non-normal distribution for the embedding layer. Instead, we propose that warming up the distribution optimization function over the course of the first epoch instead is more effective for such educational data. For example, with 1 million rows of training data divided into batches of size 100, there are 1M/100 = 10,000 batches per training epoch. One might then linearly increase the weight of the distribution component of the optimization over the course of the first 1,000 batches in the first epoch. This parameter is dependent on the specific problem and training rate selected, like many neural network parameters.

3.5 Asymmetric network structures

Autoencoder structures are typically symmetric, having a similar number of layers of similar size both before and after the central embedding layer. However, researchers have shown that an asymmetric structure leads to more robust embeddings [35]. In the case of using autoencoders to extract features for supervised classification, the network can be designed with the eventual supervised classification task in mind. If a relatively simple classifier is to be trained, the embedding should be easily interpreted (at least by a machine). Thus, it might be prudent to develop a decoder that is less complex than the encoder producing the embeddings, ensuring that the embeddings the encoder produces can be processed with a simpler function. For example, the encoder could be composed of several LSTM layers capable of recognizing detailed patterns to extract the embeddings, while the decoder might be only a few layers of fully-connected neurons. Figure 2 shows an illustration of such a network structure.

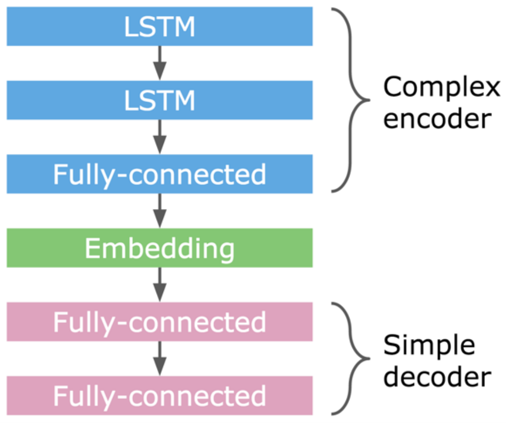

Figure 2. Asymmetric autoencoder

The asymmetric structure approach has particular importance for educational data where there is a scarcity of labeled data. A supervised classifier should find a relatively simple decision boundary in such a scenario to minimize over-fitting to the few instances of labeled data. Thus, the features extracted from autoencoders should also be relatively straightforward to decode, so that a simple decision boundary will suffice for classification. Asymmetric network structure have the potential to facilitate this goal.

3.6 Predicting future sequences

Thus far all the network structures we have discussed are designed to predict the exact sequence of student interactions that was given as input. However, the ultimate goal of these unsupervised models is not to simply compress the input, but to find a representation that can effectively be utilized as features for supervised classification. To this end, we propose neural networks that predict future steps in the sequence of student interaction data, rather than simply representing the current actions (Figure 3). In this design, the network learns to predict something new, rather than simply learning a complicated identity function. The training process is still unsupervised, since the network requires only data from the raw interaction log, yet learns features from the data that are predictive in nature.

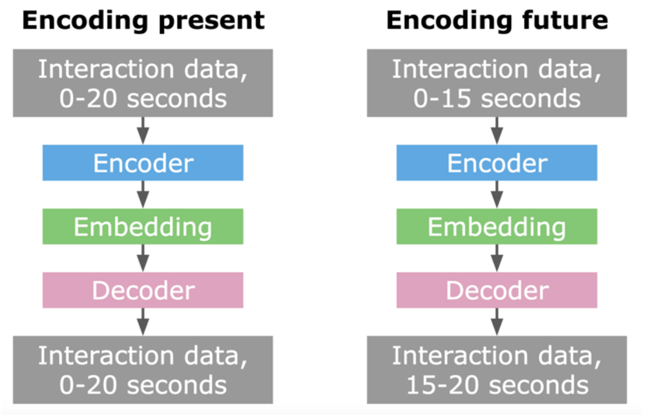
Encoding present
Interaction data, 0-20 seconds
Encoder
Embedding
Decoder
Interaction data, 0-20 seconds
Encoding future
Interaction data, 0-15 seconds
Encoder
Embedding
Decoder
Interaction data, 15-20 seconds

Figure 3. Autoencoding present vs. future interaction sequences

4. MONITORING NETWORK TRAINING

Neural networks are essentially black boxes that rarely offer interpretability. It can therefore be difficult to determine when a network is learning, if it is learning something useful, where training might be going wrong, and if so how to fix it. In this section, we discuss two common methods for monitoring the training process, and how they relate to educational data.

4.1 Training loss

The most basic measure of learning in a neural network is the training loss. The loss simply measures the difference between a network's outputs and the desired (ground truth) values. For autoencoders the measure of loss is often mean squared error, which offers relatively straightforward interpretation of network fit. However, one must be cautious when evaluating an autoencoder from a single number. The loss can decrease as a network is trained, indicating learning, but without producing meaningful embeddings. For example, as discussed earlier a VAE can find a local minimum in the loss function by generating Gaussian embeddings that don't represent students' interactions at all. An autoencoder trained with time spent per problem and score on a problem could learn to predict the mean time spent and mean score, thus decreasing loss from an original random state but without learning any meaningful embeddings that might be predictive in supervised classification. Therefore, while it is important to verify that the loss is decreasing during training (so the network is learning something), one must delve deeper to find real issues.

4.2 Visualizing network activations

Since we are using a hidden layer of the unsupervised networks as features (the embeddings) for a layer supervised learning task, it is especially important that the features learned by the deep network are suitably distributed. Embeddings are unlikely to be effective features if they are made up of neurons that are not properly trained, or have activations violating the distribution assumptions of the supervised classification methods to be employed. One important method for assessing the hidden layers of trained neural networks is visualization of some aspects of the neurons (e.g., weights, activations, or other properties).

In computer vision applications of neural networks it is common to visualize the weights of a convolution filter and find, for example, that the weights strongly resemble a rotated edge detector. Due to the nature of interaction-log data, such interpretations are not so readily available. We propose instead to randomly sample sequences of students' interaction data, feed these sequences into the autoencoder, and construct histograms of the activations of individual neurons. For the purposes of this paper, we randomly sampled 1280 instances. For autoencoder layers with more than 15 neurons, an additional visualization was created with 15 neurons chosen from uniformly distributed positions within the layer to form a representative sample for easier visualization. Full visualizations were not made for layers with more than 160 neurons as these produced intractably large images.

Using this method, we were able to identify and troubleshoot several problems with autoencoders trained on students' interaction data. Figure 4 shows activations from the embedded layer with three neurons in an example autoencoder. We used rectified linear unit (ReLU) activation in all layers of this example network except the output [28]. One issue with ReLU activation is that neurons can become stuck outputting zero [22]. The network we trained originally learned to only predict the mean of interaction inputs. The embedding layer represented in the top row of Figure 4 shows that all neurons appear to be in the stuck state, instead of displaying output values distributed across a range of different numbers. Other layers in this network reflected many stuck neurons as well.

We first addressed this by lowering the learning rate to reduce the incidence of stuck neurons, though at the expense of requiring more time to train the network. The second row of Figure 4 shows that this strategy did fix two of the neurons in the embedding. Finally, we changed the activation functions from ReLU to exponential linear units (ELUs), which are specifically designed to avoid the problem of stuck neurons frequently encountered with ReLU activation [5]. As seen in the bottom row of Figure 4, this resulted in all neurons in the embedding layer having some non-zero activations. Visual examination of the histograms from other layers revealed no stuck neurons.

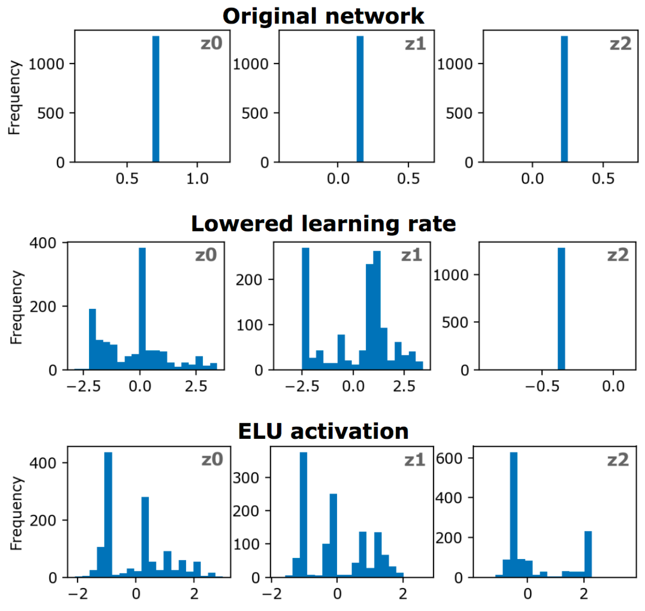

Figure 4. Histograms of neuron activations for an embedded layer and two strategies for fixing stuck neurons

Examination of neuron activations can also be helpful for verifying that a VAE is learning embeddings with a normal distribution. We enforced a normal distribution on the embedding layer in the previous example to construct a VAE, and visualized the embedding again. Figure 5 illustrates the activations of neurons in the activation layer in this VAE. The activations are now clearly far more normally distributed than before. This does not necessarily mean the VAE embedding will be more effective for feature extraction, but the visualization does show that the desired distribution is being achieved. Similarly, visualizations can be examined to determine whether operations such as batch normalization [14] are needed to shift the activations of layers to improve training.

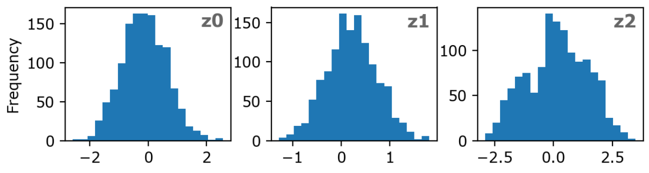

Figure 5. VAE embeddings showing activations following a (more) normal distribution

5. SUPERVISED LEARNING EXAMPLE

In this section we show a short example utilizing features from embeddings extracted with an autoencoder to detect student boredom. This is a simple and preliminary example, but demonstrates the potential of this method for future work.

5.1 Dataset

The data in this example come from student interactions with a computer-based learning environment called Betty's Brain [2]. Students interact with Betty's Brain by graphically constructing a causal map of concepts for a topic. This map becomes the "brain" of a virtual student (Betty), who then answers questions and takes quizzes. A student using the software thus assumes the role of virtual teacher, and learns by reading about a topic and then teaching it to Betty.

There were 94 students represented in these data. A total of almost 250,000 events were logged, providing a great deal of raw interaction data from which patterns may be learned with unsupervised methods. There were over 5,000 affect labels associated with these interactions. Affect labels were obtained via the Baker Rodrigo Ocumpaugh Monitoring Protocol (BROMP) [31], in which expert observers annotate student affect in real-time.

We focus on detecting student boredom, based on previous research in other learning environments showing that boredom could be detected with only a few features [18], thus making a straightforward comparison between manually-engineered high-level features and features automatically extracted with an autoencoder.

5.2 Extracting features

Features were extracted from 20-second windows of data leading up to each boredom label. Features were derived from three sources of information: the number of total interaction events, the number of times the student viewed the causal map section of the interface, and the number of times the student viewed one of the instructional materials pages (e.g., textbook). For the sake of comparison we developed an autoencoder with the same three sources of information, but without processing them into high-level features. Instead we aggregated actions at the 1-second level.

The autoencoder was trained with 20-second sequences of the 1-second data. We used 15 seconds of data as input and trained the autoencoder to predict the last 5 seconds of each sequence. Thus there were 3x15 = 45 inputs and 3x5 = 15 outputs. The network structure consisted of: LSTM (64 outputs) → LSTM (40) → Fully-connected embedding (3) → Fully-connected (8) → Fully-connected (16) → Fully-connected (15). We used 80% of the unlabeled data to train the network, and 20% as validation data to monitor training progress. The resulting embeddings are those that can be seen in the third row of Figure 4.

The traditional features (i.e. manually designed by experts) extracted for comparison were built by averaging the values of the three features across the 20-second sequence.

5.3 Supervised classification

We applied three classifiers for supervised classification, Gaussian naïve Bayes, logistic regression, and decision trees. Four-fold cross-validation was performed with different students in each fold to ensure generalization of classification results across students within the dataset.

We classified boredom vs. all other affective states in the dataset. Boredom consisted of only 4.5% of the labels. Classification accuracy was measured with AUC, since it is appropriate for datasets such as this one with imbalanced class distributions [15].

5.4 Results

The best model built using the traditional feature extraction method had AUC = .631, while the best model using automatic feature extraction method had AUC = .673. Both models were better than chance level (AUC = .5). Most notably, the model built with features extracted using the autoencoder method was better than the traditional feature method using the same raw data.

We also investigated the advantage afforded by predicting future sequences (see section 3.6). Without incorporating this method into the network structure for feature extraction, the best supervised model using autoencoder embedding features was less accurate (AUC = .643 vs. AUC = .673).

6. CONCLUSIONS

We were interested in outlining methods for applying deep learning for automatic feature extraction in educational data. Toward this goal, we detailed several different autoencoder network structures that can be trained with educational data, and students' interaction-log data in particular.

There are several ways in which training autoencoders can go wrong, including stuck neurons, networks that only predict the mean of the data, and learned embeddings that are not effective for supervised classification methods. We described methods for tweaking autoencoder structures and training to extract predictive features (section 3.6), and visualization methods to inspect neuron activations in trained networks (section 4.2). These methods allowed us to diagnose issues with an example neural network, and eventually to improve on supervised classification accuracy compared to traditional feature extraction methods.

6.1 Limitations and future work

This work is not without its limitations. The results in our example boredom detection did not greatly exceed chance levels (AUC = .673 versus .5 chance level). This is a difficult classification task, however, given that there was an imbalance of instances of boredom (4.5%). However, even if there is little improvement in accuracy, there is a great deal of potential for future work with this method in terms of transfer learning. In particular, interaction-log affect detectors have been shown to generalize poorly across populations [30]. Training a new affect detector on each new dataset is cost- and time- prohibitive, because of the large amount of manual labor required to collect affect labels. Using automatic feature extraction methods, we can train an autoencoder on a combination of existing data and data from a new population. In this way, the learned features should be representative of both populations. The existing labels from one population can then be applied to train a new supervised model with the features that are representative of both populations, in hopes that the resulting model will generalize better across both populations.

An additional limitation of this work is the fully separated unsupervised and supervised phases of the training process. This limits the ability of the neural networks to learn features that are directly related to the supervised classification goals. Instead, semi-supervised methods, in which the network learns from labeled and unlabeled data at the same time, might be a better strategy. For example, adversarial autoencoders have shown to be effective for image classification with few labels and many unlabeled instances [36].

It is our hope that this line of research will lead to improved student modeling in computerized educational environments. In particular, there is still a need for scalable student modeling methods that generalize to new populations so that educational software can reliably assess and adapt to students' affect, cognition, and behavior.

7. ACKNOWLEDGMENTS

This research was supported by the National Center for Supercomputing Applications Postdoctoral Researcher Program.

8. REFERENCES

[1] Baker, R. and Ocumpaugh, J. 2015. Interaction-based affect detection in educational software. *The Oxford Handbook of Affective Computing*. R. Calvo, S. D'Mello, J. Gratch, and A. Kappas, eds. New York: Oxford University Press. 233–245.

[2] Biswas, G., Segedy, J.R. and Bunchongchit, K. 2016. From design to implementation to practice - A learning by teaching system: Betty's Brain. *International Journal of Artificial Intelligence in Education*. 26, 1 (Mar. 2016), 350–364.

[3] Blanchard, N., Bixler, R., Joyce, T. and D'Mello, S. 2014. Automated physiological-based detection of mind wandering during learning. *Proceedings of the 12th International Conference on Intelligent Tutoring Systems (ITS 2014)* (Jun. 2014), 55–60.

[4] Bosch, N., D'Mello, S.K., Ocumpaugh, J., Baker, R.S. and Shute, V. 2016. Using video to automatically detect learner affect in computer-enabled classrooms. *ACM Transactions on Interactive Intelligent Systems (TiiS)*. 6, 2 (2016).

[5] Clevert, D.-A., Unterthiner, T. and Hochreiter, S. 2015. Fast and accurate deep network learning by exponential linear units (ELUs). *arXiv:1511.07289 [cs]*. (Nov. 2015).

[6] Corbett, A.T. and Anderson, J.R. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*. 4, 4 (Dec. 1994), 253–278.

[7] Dahl, G.E., Yu, D., Deng, L. and Acero, A. 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*. 20, 1 (Jan. 2012), 30–42.

[8] D'Mello, S., Blanchard, N., Baker, R., Ocumpaugh, J. and Brawner, K. 2014. I feel your pain: A selective review of affect-sensitive instructional strategies. *Design Recommendations for Intelligent Tutoring Systems - Volume 2: Instructional Management*. R. Sottilare, A. Graesser, X. Hu, and B. Goldberg, eds. 35–48.

[9] D'Mello, S. and Kory, J. 2015. A review and meta-analysis of multimodal affect detection systems. *ACM Computing Surveys*. 47, 3 (Feb. 2015), 43:1–43:36.

[10] D'Mello, S., Olney, A., Williams, C. and Hays, P. 2012. Gaze tutor: A gaze-reactive intelligent tutoring system. *Int. J. Hum.-Comput. Stud.* 70, 5 (May 2012), 377–398.

[11] Hinton, G.E. and Salakhutdinov, R.R. 2006. Reducing the dimensionality of data with neural networks. *Science*. 313, 5786 (Jul. 2006), 504–507.

[12] Hochreiter, S. and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation*. 9, 8 (Nov. 1997), 1735–1780.

[13] Hussain, M.S., AlZoubi, O., Calvo, R.A. and D'Mello, S.K. 2011. Affect detection from multichannel physiology during learning sessions with AutoTutor. *Proceedings of the 15th International Conference on Artificial Intelligence in Education* (Jun. 2011), 131–138.

[14] Ioffe, S. and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167 [cs]*. (Feb. 2015).

[15] Jeni, L.A., Cohn, J.F. and De la Torre, F. 2013. Facing imbalanced data–Recommendations for the use of performance metrics. *Proceedings of the 5th International Conference on Affective Computing and Intelligent Interaction* (Sep. 2013), 245–251.

[16] Jiang, D., Cui, Y., Zhang, X., Fan, P., Ganzalez, I. and Sahli, H. 2011. Audio visual emotion recognition based on triple-stream dynamic Bayesian network models. *Affective Computing and Intelligent Interaction*. S. D'Mello, A. Graesser, B. Schuller, and J.-C. Martin, eds. Berlin Heidelberg: Springer-Verlag. 609–618.

[17] Jordan, M.I. and Mitchell, T.M. 2015. Machine learning: Trends, perspectives, and prospects. *Science*. 349, 6245 (Jul. 2015), 255–260.

[18] Kai, S., Paquette, L., Baker, R., Bosch, N., D'Mello, S., Ocumpaugh, J., Shute, V.J. and Ventura, M. 2015. Comparison of face-based and interaction-based affect detectors in physics playground. *Proceedings of the 8th International Conference on Educational Data Mining (EDM 2015)* (2015), 77–84.

[19] Khajah, M., Lindsey, R.V. and Mozer, M.C. 2016. How deep is knowledge tracing? *Proceedings of the 9th International Conference on Educational Data Mining (EDM 2016)* (2016), 94–101.

[20] Kingma, D.P. and Welling, M. 2013. Auto-encoding variational bayes. *arXiv:1312.6114 [cs, stat]*. (Dec. 2013).

[21] Klingler, S., Käser, T., Busetto, A.-G., Solenthaler, B., Kohn, J., Aster, M. von and Gross, M. 2016. Stealth assessment in ITS - A study for developmental dyscalculia. *Intelligent Tutoring Systems* (Jun. 2016), 79–89.

[22] Maas, A.L., Hannun, A.Y. and Ng, A.Y. 2013. Rectifier nonlinearities improve neural network acoustic models. *Proceedings of the 2013 International Conference on Machine Learning (ICML)* (2013).

[23] Maas, A.L., Le, Q.V., O'Neil, T.M., Vinyals, O., Nguyen, P. and Ng, A.Y. 2012. Recurrent neural networks for noise reduction in robust ASR. *INTERSPEECH* (2012), 22–25.

[24] McDaniel, B.T., D'Mello, S.K., King, B.G., Chipman, P., Tapp, K. and Graesser, A. 2007. Facial features for affective state detection in learning environments. *Proceedings of the 29th Annual Cognitive Science Society* (2007), 467–472.

[25] McQuiggan, S.W., Lee, S. and Lester, J.C. 2007. Early prediction of student frustration. *Proceedings of the International Conference on Affective Computing and Intelligent Interaction* (Sep. 2007), 698–709.

[26] Metallinou, A. and Cheng, J. 2014. Using deep neural networks to improve proficiency assessment for children English language learners. *INTERSPEECH* (2014), 1468–1472.

[27] Mock, P., Gerjets, P., Tibus, M., Trautwein, U., Möller, K. and Rosenstiel, W. 2016. Using touchscreen interaction data to predict cognitive workload. *Proceedings of the 18th ACM International Conference on Multimodal Interaction* (New York, NY, USA, 2016), 349–356.

[28] Nair, V. and Hinton, G.E. 2010. Rectified linear units improve restricted boltzmann machines. *Proceedings of the 27th international conference on machine learning (ICML)* (2010), 807–814.

[29] Ng, H.-W., Nguyen, V.D., Vonikakis, V. and Winkler, S. 2015. Deep learning for emotion recognition on small datasets using transfer learning. *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction* (New York, NY, USA, 2015), 443–449.

[30] Ocumpaugh, J., Baker, R., Gowda, S., Heffernan, N. and Heffernan, C. 2014. Population validity for educational data mining models: A case study in affect detection. *British Journal of Educational Technology*. 45, 3 (May 2014), 487–501.

[31] Ocumpaugh, J., Baker, R. and Rodrigo, M.M.T. 2015. Baker Rodrigo Ocumpaugh Monitoring Protocol (BROMP) 2.0 Technical and Training Manual. *Technical Report* (2015).

[32] Paquette, L., de Carvalho, A.M. and Baker, R.S. 2014. Towards understanding expert coding of student disengagement in online learning. *Proceedings of the 36th Annual Cognitive Science Conference* (2014), 1126–1131.

[33] Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L.J. and Sohl-Dickstein, J. 2015. Deep knowledge tracing. *Advances in Neural Information Processing Systems 28* (2015), 505–513.

[34] Sønderby, C.K., Raiko, T., Maalø e, L., Sønderby, S.K. and Winther, O. 2016. Ladder Variational Autoencoders. *Advances in Neural Information Processing Systems 29* (2016), 3738–3746.

[35] Sun, Y., Mao, H., Guo, Q. and Yi, Z. 2016. Learning a good representation with unsymmetrical auto-encoder. *Neural Computing and Applications*. 27, 5 (Jul. 2016), 1361–1367.

[36] Tachibana, R., Matsubara, T. and Uehara, K. 2016. Semi-supervised learning using adversarial networks. *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)* (Jun. 2016), 1–6.

[37] Tang, S., Peterson, J.C. and Pardos, Z.A. 2016. Deep neural networks and how they apply to sequential education data. *Proceedings of the Third (2016) ACM Conference on Learning @ Scale* (New York, NY, USA, 2016), 321–324.

[38] Whitehill, J., Serpell, Z., Lin, Y.-C., Foster, A. and Movellan, J.R. 2014. The faces of engagement: Automatic recognition

of student engagement from facial expressions. *IEEE Transactions on Affective Computing*. 5, 1 (Jan. 2014), 86–98.

[39] Whitehill, J., Williams, J.J., Lopez, G., Coleman, C.A. and Reich, J. 2015. Beyond prediction: First steps toward automatic intervention in MOOC student stopout. *Proceedings of the 8th International Conference on Educational Data Mining (EDM 2015)* (2015), 171–178.

[40] Wold, S., Esbensen, K. and Geladi, P. 1987. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*. 2, 1 (Aug. 1987), 37–52.

[41] Xu, Y. and Mostow, J. 2010. Using logistic regression to trace multiple sub-skills in a dynamic bayes net. *Proceedings of the 4th International Conference on Educational Data Mining (EDM 2011)* (2010), 241–245.

[42] Zeng, Z., Pantic, M., Roisman, G.I. and Huang, T.S. 2009. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 31, 1 (2009), 39–58.