Machine-Learned or Expert-Engineered Features? Exploring Feature Engineering Methods in Detectors of Student Behavior and Affect

Anthony F. Botelho
Worcester Polytechnic Institute
Worcester, MA 01605
abotelho@wpi.edu

Ryan S. Baker
University of Pennsylvania
Philadelphia, PA 19104
ryanshaunbaker@gmail.com

Neil T. Heffernan
Worcester Polytechnic Institute
Worcester, MA 01605
nth@wpi.edu

ABSTRACT

There is a long history of research on the development of models to detect and study student behavior and affect. Developing computer-based models has allowed the study of learning constructs at fine levels of granularity and over long periods of time. For many years, these models were developed using features based on previous educational research from the raw log data. More recently, however, the application of deep learning models has often skipped this feature - engineering step by allowing the algorithm to learn features from the fine-grained raw log data. As many of these deep learning models have led to promising results, researchers have asked which situations may lead to machine-learned features performing better than expert-generated features. This work addresses this question by comparing the use of machine-learned and expert-engineered features for three previously-developed models of student affect, off-task behavior, and gaming the system. In addition, we propose a third feature-engineering method that combines expert features with machine learning to explore the strengths and weaknesses of these approaches to build detectors of student affect and unproductive behaviors.

Keywords

feature engineering, student affect, disengaged behavior, deep learning

1. INTRODUCTION

The educational data mining community has developed numerous models to detect unproductive student behaviors and affective states and study how these measures correlate with short- and long-term learning outcomes. Estimates produced by detectors of student affective states and unproductive behavior, for example, have been found to predict student standardized test scores [15], whether a student chooses to attend college [16], and whether they pursue a degree in STEM [17], and even later pursue a STEM career [11], from estimates produced from interaction logs collected as they worked on mathematics problems in seventh grade. The predictive power of these detectors along with a general desire to understand and improve the student learning process has led to a significant amount of research around developing these models.

The primary goal of this paper is to compare the two methods of generating features to predict student affect and unproductive behaviors. It is also important to consider whether some models perform better for certain types of features. Conversely, other simpler models such as a decision tree or logistic regression require feature engineering or aggregation in order to incorporate labeled and unlabeled data. Additionally, these models would be unable to easily observe unlabeled data in a semi-supervised learning manner [19].

In this work we re-develop detectors of student affective state [5], off-task behavior [15], and gaming the system [13], comparing new models to previously-developed models, to address the following research questions: 1) Which leads to better model performance (AUC ROC and Kappa), expert-engineered features or machine-learned features, for detectors of affect and unproductive behaviors? 2) Does combining expert-engineered features and machine learning-based feature generation improve model performance for detectors of affect and unproductive behaviors? 3) Does incorporating unlabeled data improve model performance for detectors of affect and unproductive behaviors?

The development of affect detectors within ASSISTments has improved in recent years. Initial work has explored the use of expert-engineered features to develop and evaluate detectors through a population validity study [12]. Recently, a deep learning approach was applied [5], utilizing the engineered features within a recurrent neural network to predict the four labels of affective state simultaneously over time.

This work utilizes two datasets[1] consisting of student interaction log data collected within the ASSISTments computer-based learning platform. The content within the system consists primarily of mathematics problems for students in grades 6-8. Students working in the system receive immediate correctness feedback on each problem with the ability to make multiple attempts, and have the ability to ask for on-

[1]All data used in this work is made openly available at the following link: `http://tiny.cc/ML_or_Expert`

demand computer-provided aid in the form of hints or scaffolded problems. These interactions and timing information are the data used to construct most of the expert-engineered features utilized in this work. The ground-truth labels of both student affect and off-task behavior were collected using quantitative field observations following the Baker Rodrigo Ocumpaugh Monitoring Protocol (BROMP) [1].

We analyze gaming the system using a data set collected via text replays [2]. The ground-truth labels of student gaming were collected using post-hoc examinations of sequences of student data following a set of criteria outlined in [2].

2. METHODOLOGY

Jiang et al. [10] compared two feature engineering methods, expert-generated and deep learning-based, for the development of affect and off-task behavior detectors within the Betty's Brain learning system. This work will test the generalizability of their findings, explore the strengths and weaknesses of each method, and explore the incorporation of unlabeled data during model training.

Each of the models described next were trained and evaluated using stratified 10-fold student-level cross validation. We stratified each fold based on the number of occurrences of positive labels of each outcome label at the student level to generate the folds of the cross validation.

2.1 Utilizing Expert-Engineered Features

The first set of models use expert-engineered features to detect each label of student affect, off-task behavior, and gaming the system using methods similar to those implemented in previous works. The expert features are first generated using the the raw action-level log data; 92 features are created for the affect and off-task data while 33 are generated for the gaming detectors in alignment with the features used in previous works. The features are generated to describe the actions that occur in 20 seconds of observation but also include neighboring actions that go beyond those 20 seconds to capture the full context. We apply a Naive Bayes classifier, a REP tree classifier and a Long-Short Term Memory (LSTM) deep learning network [9] for the gaming, off-task behavior, and affect detection tasks as these were found to work well in previous works [13][15][5].

2.2 Deep Learning Models

Unlike the expert-engineered features, the machine learned feature set uses the raw action logs of each student, ignoring the clips and clip-level features described in the previous section. For this feature set, a LSTM model is applied over the raw data to predict each outcome using a set of uninterpretable features learned within the hidden layer of the network. One potential drawback of using a LSTM model in this way is that it assumes that each timestep in the given sequence occurs at regular intervals which, of course, is not the case. Therefore, to reduce the variance of this interval, a similar practice as was applied to the affect detector model using expert-engineered features. This allows the model to divide sequences of student actions where long intervals may occur between subsequent actions; where the amount of time between two subsequent actions of the same student is greater than 5 minutes, the sequence is divided into two smaller sequences to be input into the model.

Each model utilized the same raw action-level log data that was used to generate the expert-engineered features described in the previous section. In addition to the interaction descriptors such as response correctness and whether the student requested a hint, the knowledge component associated with each problem was also included as a large 1-hot encoded vector in an effort to supply these LSTM models with the same information with which the expert-generated features had access. In addition to these described action logs, the set of features supplied to the gaming model included an additional field corresponding to the computed Levenshtein distance of each students sequence of incorrect responses (where such sequences of incorrect responses existed) within each problem as was computed for the expert-engineered features of this detector. We incorporated this feature to provide consistency in the information that is exposed to both the machine learning model and the expert feature-engineering process, although we acknowledge that the feature itself is a transformation of the raw responses (i.e. can be described as an expert feature) and was only found to be predictive of gaming the system through prior work exploring the development of expert features for this task [14].

Each of the three LSTM models created for each label of student affect, off-task behavior, and gaming the system followed the same general structure comprised of an input layer feeding into a fully-connected recurrent hidden layer of 200 LSTM nodes, and then feeding into an output layer of either 2 nodes (corresponding to a 1-hot encoded positive and negative indicator of either off-task behavior or gaming the system) or 4 nodes (corresponding to a 1-hot encoded vector with one value per observed affective state). The purpose of the hidden layer is to learn a set of 200 features from the raw action logs that are predictive of each outcome label. The commonly applied technique of dropout [18] is applied between the hidden and output layers of the network in an attempt to reduce overfitting. In all cases, a softmax activation function is applied to the output of each model and trained using multiclass cross entropy.

2.3 Machine-Learned Expert-Inspired Features

The third feature set proposed for comparison combines aspects of both expert-engineered features and machine learning. Expert features may be able to help guide a machine learning model to learn better sets of features than either method individually. In addition, since each set of expert features were presumably developed with a particular set of outcome measures in mind, such labels may also be able to help guide a machine learning model to produce meaningful, albeit uninterpretable, sets of features to detect such behaviors and affective states.

Specifically, this method utilizes a 2-step training process for a machine learning model. First, an LSTM model is built to use the raw action-level logs as input (just as was done in the previous section for the models utilizing machine learned features), but in addition to predicting each label, the model is trained to predict the set of expert-engineered features as a multi-task learning problem [7]. As the affect and off-task behavior detectors utilize the same set of action logs and expert-engineered features, we build one model to read the interaction logs. This model will predict each of the set of

Table 1: Comparison of feature sets across each of the detector models with and without co-training.

Outcome	**Feature Set**	**Model**	**AUC**	**Kappa**	Co-Trained* **AUC**	Co-Trained* **Kappa**
Off-Task	Expert	REP Tree	.734	.352	.796	.369
	Machine Learned*	LSTM	.657	.073	.657	.073
	Machine Learned Expert	REP Tree	.753	.400	.781	.405
Gaming	Expert	Naive Bayes	.774	.362	.856	.180
	Machine Learned*	LSTM	.542	-.005	.542	-.005
	Machine Learned Expert	Naive Bayes	.774	.290	.847	.327
Affect (Collectively)	Expert	LSTM	.760	.172	.777	.112
	Machine Learned*	LSTM	.695	.041	.695	.041
	Machine Learned Expert	LSTM	.662	.043	.607	.037
Confusion	Expert	LSTM	.730	.042	.762	.059
	Machine Learned*	LSTM	.666	.042	.666	.042
	Machine Learned Expert	LSTM	.609	.01	.596	.018
Engaged Concentration	Expert	LSTM	.775	.281	.791	.289
	Machine Learned*	LSTM	.713	.210	.713	.210
	Machine Learned Expert	LSTM	.671	.188	.611	.090
Boredom	Expert	LSTM	.775	.148	.783	.178
	Machine Learned*	LSTM	.690	.137	.690	.137
	Machine Learned Expert	LSTM	.677	.041	.613	.005
Frustration	Expert	LSTM	.761	.054	.772	.050
	Machine Learned*	LSTM	.713	.060	.713	.060
	Machine Learned Expert	LSTM	.689	.019	.609	.026

*All co-training used an LSTM model. Models using the machine learned features all used co-training.

expert-engineered features corresponding with the given set of actions, the affective state label, and the off-task behavior label simultaneously. Similarly, for the gaming detector, the raw actions are supplied as input to a LSTM model that predicts both the set of expert-engineered features and the gaming labels. In this way, the hidden layer of the respective models is regularized to learn a set of features that is both able to construct the set of expert features (although likely with some error) as well as predict the outcome labels for which the features are intended.

Once these LSTM models are trained, the hidden layer is extracted and used as the third and final set of features compared in this work. This feature set, referred to as "machine-learned expert-inspired" features, is then supplied as input to each of the respective models used in previous work.

2.4 Exploring the use of Co-Training

In this work, we use the term "co-training" to describe the use of both labeled and unlabeled data to inform model estimates and the methods differ from that of other works describing co-trained models [3]. Given the nature of the observation-based label collection procedure, not all examples in our data (whether considering actions or clips) has an associated affect or behavior label. While there are several modeling methods that exist to incorporate this unlabeled data into each model, the already-described LSTM model inherently allows for this co-training to occur given its sequential structure. The model uses the current supplied timestep along with a learned-aggregation of previous time steps in order to better inform each prediction.

3. RESULTS AND DISCUSSION

We compare the results of each set of models within each of the three outcome measures of affect, off-task behavior, and gaming behavior using the metrics of AUC ROC and Cohen's Kappa. In the case of affect, AUC ROC is calculated using a multi-class variant of the metric [8], while Kappa is calculated as multi-class Cohen's Kappa, while the models of off-task behavior and gaming use an optimized form of Kappa by learning an optimal decision (0,1) threshold using the training set of each respective fold within the cross validation. Higher values of either metric are indicative of higher model performance with AUC values at 0.5 and Kappa values at 0 indicating chance performance.

The results of each model is reported in Table 1. Compared to the models utilizing machine learned features, the expert-engineered features lead to notably higher performance across all the outcome labels in regard to both AUC and Kappa. When comparing the performance of the models using the machine-learned expert features (our proposed third feature set), the difference in performance is not as dramatic, but leans in favor of the expert-engineered features leading to superior models. By contrast, the machine learned expert features led to models that outperform those using expert-engineered features in regard to off-task behavior in terms of both AUC and kappa and is equal in regard to detecting gaming in terms of AUC, but appears to perform less well in detecting student affect.

Despite the small number of cases where the models trained from expert-engineered features did not outperform the others in either AUC or Kappa, these models exhibited consistently high performance in both metrics across all outcomes. It can further be concluded that co-training led to higher AUC than the non-co-trained variant of each detector. This, however, was not always the case in regard to Kappa, where the co-trained models of gaming and affect using expert-engineered features exhibited notably lower values.

4. FUTURE WORK

Among the detectors, it was found in some cases that there was disagreement between the metrics used; the co-trained model of affect exhibited higher AUC than the non-co-trained model, but a lower value of Kappa. Previous work has explored this case across several commonly-applied metrics [4], but further work is needed to further explore and leverage modeling techniques to produce detectors that exhibit high performance across all these metrics.

The use of the highest-performing models across each of these detectors can also be used in future research to study other aspects of student learning. This extends beyond the already-discussed application in predicting student longer-term outcomes, and includes the study of other aspects such as the dynamics and chronometry, studied using affect detectors in prior work [6].

5. CONCLUSIONS

This work investigates whether expert-engineered features lead to higher performing detectors of student affect and disengaged behavior as compared to using automatically-distilled features learned through a machine-learning approach. We found that the use of expert features led to the most consistently high-performing models. Using co-training led to even better models in most cases.

The use of expert-engineering to develop features, while perhaps more difficult in regard to time, effort, and likely cost, does appear to lead to greater benefits than simply applying a machine learning model to automatically distill features from the raw data, based on the results found in this work.

6. ACKNOWLEDGMENTS

We thank multiple current NSF grants (ACI-1440753, DRL-1252297, DRL-1109483, DRL-1316736, DGE-1535428, OAC-1724889,OAC-1636782 & DRL-1031398) , the US Dept. of Ed (IES R305A180401, R305A120125 & R305C100024 and GAANN), and the ONR.

7. REFERENCES

[1] R. Baker, J. Ocumpaugh, and J. Andres. Bromp quantitative field observations: A review. *R. Feldman (Ed.) Learning Science: Theory, Research, and Practice*, In Press.

[2] R. S. Baker, A. T. Corbett, and A. Z. Wagner. Human classification of low-fidelity replays of student actions. In *Proceedings of the Educational Data Mining Workshop at the 8th International Conference on Intelligent Tutoring Systems*, volume 2002, pages 29–36, 2006.

[3] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.

[4] N. Bosch and L. Paquette. Metrics for discrete student models: Chance levels, comparisons, and use cases. *Journal of Learning Analytics*, 5(2):86–104, 2018.

[5] A. F. Botelho, R. S. Baker, and N. T. Heffernan. Improving sensor-free affect detection using deep learning. In *International Conference on Artificial Intelligence in Education*, pages 40–51. Springer, 2017.

[6] A. F. Botelho, R. S. Baker, J. Ocumpaugh, and N. T. Heffernan. Studying affect dynamics and chronometry using sensor-free detectors. pages 157–166, 2018.

[7] R. Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

[8] D. J. Hand and R. J. Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine learning*, 45(2):171–186, 2001.

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

[10] Y. Jiang, N. Bosch, R. S. Baker, L. Paquette, J. Ocumpaugh, J. M. A. L. Andres, A. L. Moore, and G. Biswas. Expert feature-engineering vs. deep neural networks: Which is better for sensor-free affect detection? In *International Conference on Artificial Intelligence in Education*, pages 198–211. Springer, 2018.

[11] J. Makhlouf and T. Mine. Predicting if students will pursue a stem career using school-aggregated data from their usage of an intelligent tutoring system. In *Educational Data Mining 2018*, pages 533–536, 2018.

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

[13] L. Paquette, R. S. Baker, A. de Carvalho, and J. Ocumpaugh. Cross-system transfer of machine learned and knowledge engineered models of gaming the system. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 183–194. Springer, 2015.

[14] L. Paquette, A. de Carvahlo, R. Baker, and J. Ocumpaugh. Reengineering the feature distillation process: A case study in detection of gaming the system. In *Educational data mining 2014*. Citeseer, 2014.

[15] Z. A. Pardos, R. S. Baker, M. O. San Pedro, S. M. Gowda, and S. M. Gowda. Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, pages 117–124. ACM, 2013.

[16] M. O. Pedro, R. Baker, A. Bowers, and N. Heffernan. Predicting college enrollment from student interaction with an intelligent tutoring system in middle school. In *Educational Data Mining 2013*, 2013.

[17] M. O. San Pedro, J. Ocumpaugh, R. S. Baker, and N. T. Heffernan. Predicting stem and non-stem college major enrollment from middle school interaction with mathematics educational software. In *Educational Data Mining 2014*, pages 276–279, 2014.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[19] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.