Embedding navigation patterns for student performance prediction

Ekaterina Loginova
Ghent University
ekaterina.loginova@ugent.be

Dries F. Benoit
Ghent University
dries.benoit@ugent.be

ABSTRACT

Predicting academic performance using trace data from learning management systems is a primary research topic in educational data mining. An important application is the identification of students at risk of failing the course or dropping out. However, most approaches utilise past grades, which are not always available and capture little of the student's learning strategy. The end-to-end models we implement predict whether a student will pass a course using only navigational patterns in a multimedia system, with the advantage of not requiring past grades. We experiment on a dataset containing coarse-grained action logs of more than 100,000 students participating in hundreds of short course. We propose two approaches to improve the performance: a novel encoding scheme for trace data, which reflects the course structure while remaining flexible enough to accommodate previously unseen courses, and unsupervised embeddings obtained with an autoencoder. To provide insight into model behaviour, we incorporate an attention mechanism. Clustering the vector representations of student behaviour produced by the proposed methods shows that distinct learning strategies specific to low- and high- achievers are extracted.

Keywords

learning strategies, academic performance prediction, navigational patterns, mooc, lstm, autoencoder, learning management systems

1. INTRODUCTION

A large amount of trace data about learner behaviour has recently become available from online learning environments [30]. Hence, it is now possible to improve the delivery, assessment and intervention quality using data mining techniques, giving rise to technology-enhanced learning. Educators are especially interested in receiving early alerts when students are at risk of failing the course or dropping out. With these alerts, timely intervention can be organised. To estimate this risk, machine learning classification models are built to predict student performance based on their interaction with the course content. There are several ways to define student performance: it can be a binary or a multi-level grade on either the next exercise or the entire course. This study focuses on a binary grade for the given course (fail or pass), a common scenario on student performance prediction [15, 28, 27].

Traditionally, the primary source of information for this task is past academic performance records [31]. They might include the grades for the previously taken courses or intermediate test scores. However, apart from those, online learning platforms also provide information about other types of interaction between students and the content. Depending on the technical implementation and medium, it can be fine-grained click-stream video data, the text of discussion forum messages, or more coarse-grained information such as whether a person liked a video or performed a search query. Our models operate on such coarse-grained sequential records of interacting with different content types on the online learning platform and does not rely on previous grades. The motivation for it is three-fold. First, the past scores might not be available, or it can be time- and effort-consuming to provide them. For example, grading an essay usually requires a specialist, which is not scalable for MOOCs. Second, by focusing on interaction with the content, such as video views, we obtain the representation of student behaviour that is more likely to capture their learning strategies. In other words, we can discover whether they prefer a specific medium or actively interact with other students in online discussions. Third, if we work with navigation patterns, the resulting model has the potential to inform a recommendation system that would nudge a struggling student in the right direction. For instance, when students explore the platform, we can automatically recommend the next learning item to interact with (e.g., a video or a reading material they might find useful or interesting).

This study applies recurrent neural networks for academic performance prediction in short online courses. The approach we describe works without manual feature engineering or information about previous scores. In contrast to most works in the area of course grade prediction, our models operate instead on raw sequences of multiple-type interactions with the content (video view/like, discussion message, search request, exercise attempt). We propose several encoding schemes for action logs and demonstrate that they can increase classification performance. Among them, we introduce a flexible scheme that reflects relative progress within the course while being independent of its length. We also use autoencoders to extract student representation in an unsupervised way. Vector representations of student behaviour produced by different methods are clustered to compare how well they indicate (un)successful learning strategies. In order to illuminate the inner workings of the model, we provide cluster visualisations and experiment with an attention mechanism.

Thus, our research questions are the following. **RQ1**: Can we capture the information about course structure in the action sequences in a flexible way (that can be extended dynamically when new exercises or longer courses are added)? **RQ2**: Can we improve predictive performance by pretraining unsupervised embeddings (using autoencoder models)? **RQ3**: Can we improve predictive performance further by adding attention to an autoencoder?

2. RELATED WORK

While working with educational trace data, there are various ways to extract features from the raw sequences. The easiest and perhaps the most popular approach is to count the number of actions – despite its simplicity, it is a solid baseline, used even in the most recent studies [27], [28], [21].

Unfortunately, by aggregating the data in such a way, we ignore the time delay between actions and the information about their order. A common way to engineer additional, more time-sensitive features from trace data is to represent student behaviour in chunks, called studying sessions. For instance, we can define the sequence of actions as a session if less than 15 minutes passed between actions. The threshold is usually determined heuristically [20]. However, the need to choose this threshold manually is a clear drawback. Moreover, additional manual feature engineering is often required to aggregate features over sessions (e.g., experts have to define what qualifies as session intensity).

Furthermore, even though the described measures are sound and easy to use, their high volume does not necessarily contribute to the high quality of learning. Neither does it allow actionable insight beyond relatively trivial advice to spend more time in the system. As justly noted by the learning analytics community, it is the specific learning strategies adopted by individual students that are important [12].

Therefore, various approaches based on deep learning were proposed to overcome these limitations. The sequential nature of the data lends itself well to the use of recurrent neural networks (RNNs) [37]. The pioneering Deep Knowledge Tracing (DKT) model applied RNNs and its variations to the history of students' answers to predict whether they will answer the next exercise correctly [26]. One of the benefits of such neural architectures is that they can complement the educational theories proposed by human experts with insights obtained in the bottom-up, data-driven way [23]. Another benefit is that end-to-end models adapt to new domains easier and are more cost-efficient.

The success of DKT led to a new strand of research. Aiming to get rid of its simplifying assumptions, such as disregard to skill interaction or exercise text, researchers developed more advanced neural models for knowledge tracing [8].

In contrast to many grade prediction approaches such as [18], our approach does not require knowing previous academic performance or past grades. Instead, we investigate whether a binary course grade can be predicted from the trace data alone, with no intermediate exercise scores. Thus, an advantage of our approach is that we can detect low- and high-achievers without assessing the correctness of students' answers. The benefit is especially important for courses that include open-answer questions since those usually require costly human experts to be graded reliably (for example, essays in humanities subjects).

Our models work on raw sequences actions without aggregation of count or session variables. Recent studies on using RNNs on click-stream data [22], [17], [7], [16] and [18] are conceptually close to our approach in terms of using RNNs to work with sequences of actions. However, all of them but [18] operate only on video interaction and exercise answer features, whereas we also include search queries and discussion messages. [18] does not explore the autoencoders or attention and does not investigate the extracted student representation. Moreover, in most of them, aggregation still happens: [17] uses cumulative counts, [16] – weekly snapshots and in [22] interaction features are binary per item, while we feed the raw sequences as an input to predictive models. Besides, their task is to predict the next exercise response while we predict the overall course success.

Concerning the use of deep learning for unsupervised feature extraction, only a few recent publications have explored it [7]. For example, autoencoders, a popular approach in natural language processing [35], have only recently entered the educational data mining field [36]. Motivated by this, we extend recent end-to-end approaches to feature engineering on trace data by using autoencoders (including an attentional one) to embed student behaviour.

Regarding the character of the trace data used, we operate on short courses which contain multimedia content (such as videos and discussion messages; details are provided in section 3). The dataset also addresses the variability across a range of subjects [24]. Moreover, it allows us to showcase the methods in a real-life scenario, as the data is collected from a commercial educational platform. It should be noted that only coarse-grained trace data is available, i.e. there are no details of interaction with videos, such as replays. To enrich the data representation without changes to the original platform, we introduce several encoding schemes that capture the relative progress within the course.

Despite their promising results, the state-of-the-art deep learning methods are black-box models, which renders the interpretation of the prediction making process and incorporation of domain knowledge far from straightforward [10]. Such lack of interpretability can seriously hinder the adoption of otherwise efficient models in decision-critical domains such as education, as stakeholders cannot control or assess the fairness of the process. In order to illuminate how the model makes a decision, we can cluster the produced student behaviour representations or investigate attention heatmaps. In spite of the apparent success of attention mechanisms in

Table 1: An example data entry for student behaviour in a course.

user_id	77
course_id	60
avg_session_duration	584
num_sessions	5
avg_session_intensity	2.6
session_frequency	0.24
actions	[('video_view' '2015-11-13 15:33'), ...]
views	4
searches	0
messages	0
video likes	3

natural language processing [2], few researchers have utilised them for academic performance prediction so far [25].

While our primary focus is on predicting academic performance, it is also necessary to provide an insight into the learning strategies of students [12]. While this concept is reminiscent of learning styles [9], it avoids their heavily criticised assumptions [29] by performing analysis bottom-up based on raw data, instead of fitting the students in a rigid framework. One way to investigate different learning strategies is to cluster students based on their trace data. The resulting clusters implicitly classify students according to how they receive and process information or whether they are high- versus low-achievers. The obtained cluster assignment allows us to improve personalisation mechanisms since we can now use the student's preferred mode of interaction (for example, adjust the proportion of videos versus readings based on how much of a visual learner the student is). Previous research in this field relied on Hidden Markov Models, pattern mining, or Levenshtein distance between sequences [6, 13]; on numerical features, Partitioning Around Medoids [11] or k-Means are used. This study clusters the embeddings produced by predictive models and autoencoders. We expect students with similar learning strategies to appear in the same clusters. By aligning these clusters with academic performance scores, we could distinguish strategies typical for low- and high- achievers.

3. DATA & FEATURES

3.1 Data

The dataset for this study consists of student trace data extracted from an online educational platform for secondary school students (12-18 years old), with a focus on mathematics and Dutch language courses. The dataset contains interaction logs for 44 333 students and 467 short courses (177 873 students-course tuples) from 20 September 2012 to 8 August 2020. Each course includes several lessons with associated videos, discussion threads and exercises (mostly multiple choice).

The target variable is the score that the student obtains for the course. We converted the original score (from 0 to 100) into a binary variable ($I(score > 50)$), as we are interested in patterns corresponding to general success or failure in the course. Thus, for a given student-course tuple, we need to predict 0 if the student is likely to fail the course and 1 otherwise.

For this binary classification case, there is a noticeable class imbalance: there are 124 248 (70%) instances in class 1 and 53 625 (30%) in class 0. It is also important to note that the courses on the platform are rather short compared to most datasets in the field: the median number of actions per student-course tuple is 8, and the median duration of a course is approximately 14 minutes. In comparison, [7] use information about 44 920 students participating in a 4-month course focused on a single subject, with 20 interaction features.

3.2 Features

We distinguish three ways to engineer features (see Table 1 for the example data entry). Count and session features are traditional predictors. Sequences of actions are also often aggregated per timestep (e.g. a chapter in the course) instead of being used as-is. In contrast, we use the sequences in their raw, original format as input for neural models. Such a general data format can be applied in any online course with minimal technical requirements for tracking student behaviour, which benefits smaller learning platforms. More precisely, the features we use are as follows:

1. counts of actions X_c. For example, the student with id 77 watched four videos in a course with id 60. These features include the number of: video views, video likes, messages posted in forums, search queries, questions attempted (without making a distinction between correct or incorrect answer).

2. manually engineered session features X_s. We experimented with multiple threshold timeout values (as we did not have access to login and logout timestamps for students), settling on 15 minutes. We then aggregated statistics about individual studying sessions, resulting in 4 features: average session duration in seconds, number of sessions, session frequency (the ratio of number of sessions to the course duration in hours) and session intensity (average number of actions per session). This is a typical set of features engineered in similar studies [32], which we use as a benchmark.

3. raw action sequences X_a. There are five possible actions: a video view, a video like, a discussion forum message post, a search request for a term, and an attempt to answer an exercise (without the indicator of whether the answer was correct).

Due to the short duration of courses, encoding with just five types of actions without any additional information leads to low variability in data: out of 177 873 sequences, only 10 058 are unique. As a consequence, the same behaviour pattern could correspond to both passing and failing the course. To overcome this issue, we consider several ways to encode additional information in elements of the sequence. We list them by their generalisability, from least to most.

Concatenating global content item id and the corresponding actions. For instance, for a video with id 12, we would encode the action as "video_view_12" (we give examples for videos, but the idea transfers directly to other content items as well). We can also encode items associated with the video:

answering the exercise about this video would be encoded as "exercise_answer_12". The downside is that both these approaches do not generalise to new courses: we have to assume that the number of either courses or content items is fixed. Otherwise, a model needs to be retrained. It is a common problem also found in such popular approaches as DKT [33]. Besides, this approach quickly leads to an inflated vocabulary, making it computationally inefficient if one would use the bag-of-words approach (hence it is not featured in Table 2 for machine learning models, only for recurrent networks).

Encoding actions using local content item ids, preserving their relative order in the course. In other words, if the video with id 12 is the first video in a given course, then we encode the action as "video_view_1". This scheme has the potential for interesting insights into student behaviour. For example, we can see whether consequently watching the videos contributes a lot to better performance. Besides, we can gauge the engagement and background knowledge by checking whether the student immediately watched later videos. Regarding the disadvantages, the applicability to new courses is still limited: the number of videos should be the same or less than in the courses we have seen before. Using the out-of-vocabulary token or setting the maximum possible number of videos high is the simple workaround. However, a more flexible solution might be required – for which we suggest the progress percentage encoding.

Encoding the rounded percentage of the total number of videos in the course. For instance, if there are five videos in total, then the view of the second one will be represented as "video_view_40" (40%). Even though we can no longer focus on the exact content item as if the case with the other encoding schemes, we can still potentially recommend the section of the course to revise. This is the most flexible way, as it scales to new courses (of arbitrary length, with previously unseen exercises) as well.

We provide experimental results using the four proposed encoding schemes (raw X_a, global content id X_{a-gid}, local content id X_{a-lid} and progress percentage id X_{a-pid}, respectively) in section 5. To each of the above schemes, we can also add the difference in seconds with the previous action if the timestamps are available.

4. MODELS

We trained three types of classification models on numerical count features X_c and session features X_s: Logistic Regression, Decision Tree and Random Forest. For sequence data X_a, we used two popular variations of RNN – Long-Short Term Memory (LSTM) [14] and Gated Recurrent Unit (GRU) [4]. We have experimented with convolutional neural networks, but their performance was lower than that of recurrent ones. As such, for the sake of brevity, we do not focus on them in this study.

On a general level, neural classification models embed action sequences and pass them through recurrent layers to the final feed-forward layer(s) with sigmoid activation. It produces a probability of success which is then converted into a classification prediction score. A recurrent neural network takes a sequence of vectors $\{x_t\}_{t=1}^T$ (T is the number of timesteps) as an input and maps them to an output sequence $\{y_t\}_{t=1}^T$ by calculating hidden states $\{h_t\}_{t=1}^T$ which encode past information that is relevant for future predictions. LSTM uses a more elaborate structure in each of the repeating cells, allowing it to learn long-term dependencies. It includes so-called forget, input and output gates, which control what information to retain and pass to the next step. GRU simplifies the cell by combining the forget and input gates into a single update gate and merges the cell state and hidden state.

An encoder-decoder framework uses an RNN to read a sequence of vectors $\{x_t\}_{t=1}^T$ into another, context vector c in an auto-regressive fashion. If we set the desired output sequence equal to the input one – so that the goal becomes the reconstruction of the original data – we obtain an autoencoder. Then, the context vector, if its dimensionality is chosen to be lower than that of the input, will contain a denoised representation of the data that can be used in other models. This way, an autoencoder allows us to learn efficient data representation in an unsupervised manner.

Bahdanau attention mechanism allows the network to focus on certain parts of the input [2]. It is achieved by computing the context vector as a weighted sum of vectors produced by the encoder. The weights are learnt by a feed-forward neural network jointly with the rest of the model. Transformer model is a recent competitive alternative to the encoder-decoder framework [34] which foregoes the recurrent cells in favour of stacked self-attention and feed-forward layers.

5. EXPERIMENTS

We apply machine learning models on count and session features and compare them with deep learning ones on sequences of actions (with different encoding schemes used, as outlined above). Besides, we embed action sequences using an LSTM autoencoder and use its output X_{auto} as input for the classification models.

Neural network models were implemented using Keras [5] with Tensorflow backend [1] and machine learning ones with sklearn [3]. The parameters were optimised using the grid-search with stratified 10-fold cross-validation (5-fold for neural models). For recurrent neural networks, we checked the following parameter values: 32/64/128 recurrent units, 32/64/128 hidden units in feed-forward layers, 32/64/128 embedding dimensions. The maximum sequence length was set to 50. The models were trained with binary cross-entropy loss, using the Adam optimiser [19], early stopping and learning rate reduction on a plateau.

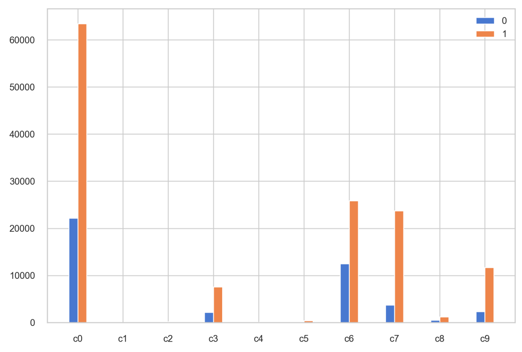
5.1 Classification

For the binary classification task, the cross-validation ROC AUC scores are presented in Table 2. Inspecting the table, we can conclude that end-to-end models perform at least as well as the ones using manually engineered features, even when the length and variability of actions sequences are limited.

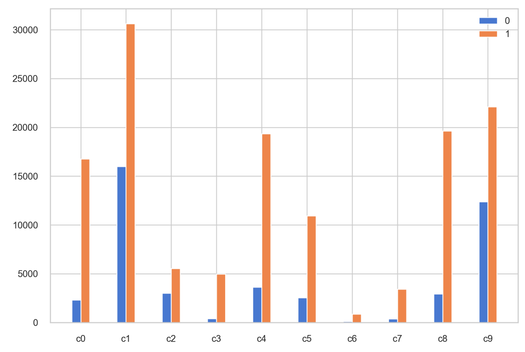
Concerning **RQ1**, the contribution of different encoding schemes, we can see that global content id encoding performs the best. However, as mentioned above, it does not scale to new content items. However, using the percentage encoding scheme

Table 2: Cross-validation ROC AUC scores for classification models. Input features: count features X_c, session features X_s, action sequences X_a with encoding scheme variations (no id X_a, global content id X_{a-gid}, local content id X_{a-lid}, progress percentage id X_{a-pid}), actions embeddings by LSTM autoencoder X_{auto}, actions embeddings by an attentive (Bahdanau) LSTM autoencoder X_{auto_B}.

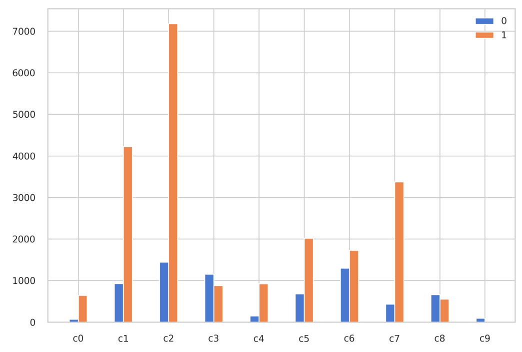
	X_c	X_s	X_{a-lid}	X_{a-pid}	X_{auto}	X_{auto_B}
Logistic Regression	0.62	0.62	0.69	0.67	0.77	0.76
Random Forest	0.73	0.81	0.81	0.81	0.82	0.81
Decision Tree	0.73	0.80	0.79	0.80	0.80	0.80
	X_a	X_{a-gid}	X_{a-lid}	X_{a-pid}	X_{auto}	X_{auto_B}
LSTM	0.73	0.88	0.82	0.81	0.82	0.83
GRU	0.73	0.87	0.81	0.81	0.83	0.82



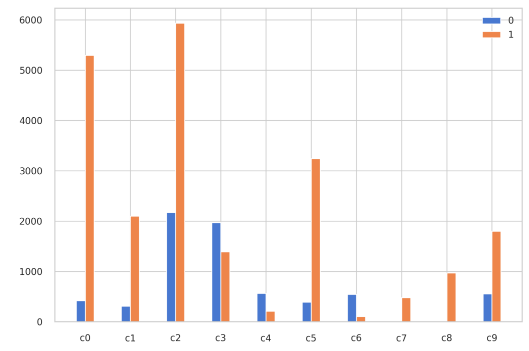
(a) X_c (count features)



(b) X_s (session features)



(c) X_a (LSTM embeddings on action data)



(d) X_{a-pid} (LSTM embeddings on action data with progress percentage id)

Figure 1: Distribution of target variable (pass/fail score) over the K-Means clusters based on different data representations (with Euclidean distance; K of 3, 5, 10 and 20 were tried). LSTM embeddings distinguish better between high- and low-achievers, producing clusters that clearly correspond to one class more than the other, while for traditional count (a) and session (b) data, most of the clusters contain a mix of both classes. For LSTM embeddings on action data (c), clusters appear that contain more failing students than passing – and are thus more useful for early warning systems. The effect is even more pronounced if we use progress percentage encoding (d).

also gives a steady increase in ROC AUC score: from 0.73 to 0.81 for Random Forest, where we use count features, and similarly for recurrent networks, where we use sequences. Using unsupervised embeddings obtained with an autoencoder (the focus of **RQ2**), we gain an improvement as well: from 0.73 to 0.82 for both machine and deep learning models.

Contrary to our expectations for **RQ3**, adding attention did not significantly improve the results. We experimented with including an attention mechanism directly into the classification models, both Bahdanau and Transformer. It can be done, for example, by training a simple Transformer, then using the average of its encoder's hidden states as an input to a classification model on top. Unfortunately, those modifications did not influence performance in our experiments, in contrast to [25]. We hypothesise that the features might be too coarse-grained and the sequences too short to take full advantage of the technique; there are also no skill labels that would provide the hierarchical structure that attention mechanisms can reflect. However, using attention allows us to produce visualisations that a first step to understanding the decision-making process of the neural network. Thus, we provide the attention scores from a Bahdanau-attention classification model below for illustration purposes.

To gain more actionable insight from our models, we also investigated the predictive performance on partial action sequences. Being able to predict early that a student will fail the course would allow sending a timely alert to the educator, signalling the need for intervention. Hence, we explored how the performance of the models changes when only the first N actions are available. For these experiments, to ensure that the model does not have full information, we only used courses with more than five questions and more than eight actions. As depicted in Figure 2, sequential data lead to higher scores than count features and proposed encoding schemes outperform raw action sequences.

5.2 Clustering & Visualisation

We clustered the student-course tuples based on different data representations using k-Means and plotted the distribution of the target class over these clusters (see Figure 1d) to investigate how the models distinguish between learning strategies of low- and high-achievers. For traditional count and session data, most of the clusters are not easily interpretable, as they contain a mix of both classes, roughly following the target label distribution. For instance, when using count features, almost all student-course pairs are in just three clusters, so there is little distinction between learning strategies. It should be noted that even though some clusters appear empty, in fact they still contain a very small number of students.

When we increase the representation's complexity, the extracted groups are more distinct. The distribution of high- and low-achievers in them shifts so that clusters with the prevalence of a single class emerge. The improvement is even more noticeable with progress percentage encoding: more clusters are extracted where one class is prevalent. For LSTM embeddings on action data, clusters appear that contain more failing students than passing, signalling that this is an unsuccessful strategy (clusters 4 and 6 on 1). This information is vital for early warning systems. The effect is even more pronounced if we use the progress percentage encoding, which encourages the application of these schemes for distinguishing between successful and unsuccessful learning strategies.

Another way to shed light on the prediction process of a neural network is to visualise attention heatmaps, where higher scores correspond to actions in the sequence that are more important for the classification decision (Figure 3).

Limitations & Future work. A wide range of topics covered in these courses might influence the performance, as different subjects are likely to demand different behaviour patterns to successfully pass the course (e.g., humanities versus technical subjects). We plan to investigate them separately and include information about exercise content. Finally, we plan to train a recommendation system informed by the extracted learning strategies to aid navigation.

6. CONCLUSION

We show that it is possible to predict whether students will pass the course using only their navigation pattern sequence in the online learning platform, without information about past grades. The findings of our study suggest that features extracted with deep learning are efficient even if the courses are extremely short, cover multiple different subjects, and only a limited number of interaction types is available.

We propose a flexible way to increase the classification performance with minimal preprocessing of the raw sequences of actions extracted from the learning platform required. A novel and relatively simple percentage progress encoding scheme is introduced which captures the course structure while scaling well to the new data. It results in an improvement of almost 10% in ROC AUC score. We also demonstrate a positive effect of using pre-trained unsupervised embeddings obtained with autoencoders (up to 15% improvement when using in machine learning models, compared to traditional features). We cluster resulting embeddings to show that using action sequences has more potential for distinguishing between strategies specific to high and low achievers than simple count or session features. It is possible to visualise attention heatmaps and see the contribution of individual actions to the classification decision to interpret retrieved strategies.

Our research supports decision-makers, as it allows detecting (un)successful students from their navigation patterns alone, without having to grade intermediate exercises. The action sequences corresponding to high achievers can be used to inform learning design patterns and recommendation systems in a more meaningful way than the standard count features. As the proposed models are shown to outperform several baselines on extremely short, incomplete action sequences, they allow us to intervene early if a student begins to follow a trajectory associated with a lower chance of success.

7. REFERENCES

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser,

M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. 2015.

[3] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[4] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[5] F. Chollet et al. Keras. https://keras.io, 2015.

[6] M. C. Desmarais and F. Lemieux. Clustering and visualizing study state sequences. In S. K. D'Mello, R. A. Calvo, and A. Olney, editors, *Proceedings of the 6th International Conference on Educational Data Mining, Memphis, Tennessee, USA, July 6-9, 2013*, pages 224–227. International Educational Data Mining Society, 2013.

[7] M. Ding, K. Yang, D. Yeung, and T. Pong. Effective feature learning with unsupervised learning for improving the predictive models in massive open online courses. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge, LAK 2019, Tempe, AZ, USA, March 4-8, 2019*, pages 135–144. ACM, 2019.

[8] X. Ding and E. C. Larson. Why deep knowledge tracing has less depth than anticipated. 2019.

[9] J. Feldman, A. Monteserin, and A. Amandi. Automatic detection of learning styles: state of the art. *Artif. Intell. Rev.*, 44(2):157–186, 2015.

[10] R. C. Fong and A. Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 3449–3457. IEEE Computer Society, 2017.

[11] D. Furr. Visualization and clustering of learner pathways in an interactive online learning environment. In M. C. Desmarais, C. F. Lynch, A. Merceron, and R. Nkambou, editors, *Proceedings of the 12th International Conference on Educational Data Mining, EDM 2019, Montréal, Canada, July 2-5, 2019.* International Educational Data Mining Society (IEDMS), 2019.

[12] D. Gašević, S. Dawson, and G. Siemens. Let's not forget: Learning analytics are about learning. *TechTrends*, 59(1):64–71, 2015.

[13] N. Gitinabard, T. Barnes, S. Heckman, and C. F. Lynch. What will you do next? A sequence analysis on the student transitions between online platforms in blended courses. 2019.

[14] S. Hochreiter and J. Schmidhuber. LSTM can solve hard long time lag problems. In M. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2-5, 1996*, pages 473–479. MIT Press, 1996.

[15] B. Jeon, E. Shafran, L. Breitfeller, J. Levin, and C. P. Rosé. Time-series insights into the process of passing or failing online university courses using neural-induced interpretable student states. 2019.

[16] H. Karimi, T. Derr, J. Huang, and J. Tang. Online academic course performance prediction using relational graph convolutional neural network. In A. N. Rafferty, J. Whitehill, C. Romero, and V. Cavalli-Sforza, editors, *Proceedings of the 13th International Conference on Educational Data Mining, EDM 2020, Fully virtual conference, July 10-13, 2020.* International Educational Data Mining Society, 2020.

[17] T. Käser and D. L. Schwartz. Exploring neural network models for the classification of students in highly interactive environments. 2019.

[18] B. Kim, E. Vizitei, and V. Ganapathi. Gritnet: Student performance prediction with deep learning. 2018.

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. 2015.

[20] V. Kovanovic, D. Gasevic, S. Dawson, S. Joksimovic, R. S. Baker, and M. Hatala. Penetrating the black box of time-on-task estimation. In J. Baron, G. Lynch, N. Maziarz, P. Blikstein, A. Merceron, and G. Siemens, editors, *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge, LAK '15, Poughkeepsie, NY, USA, March 16-20, 2015*, pages 184–193. ACM, 2015.

[21] B. Mbouzao, M. C. Desmarais, and I. Shrier. A methodology for student video interaction patterns analysis and classification. In M. C. Desmarais, C. F. Lynch, A. Merceron, and R. Nkambou, editors, *Proceedings of the 12th International Conference on Educational Data Mining, EDM 2019, Montréal, Canada, July 2-5, 2019.* International Educational Data Mining Society (IEDMS), 2019.

[22] K. Mongkhonvanit, K. Kanopka, and D. Lang. Deep knowledge tracing and engagement with moocs. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge, LAK 2019, Tempe, AZ, USA, March 4-8, 2019*, pages 340–342. ACM, 2019.

[23] S. Montero, A. Arora, S. Kelly, B. Milne, and M. Mozer. Does deep knowledge tracing model interactions among skills? 2018.

[24] B. Motz, J. Quick, N. L. Schroeder, J. Zook, and M. Gunkel. The validity and utility of activity logs as a measure of student engagement. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge, LAK 2019, Tempe, AZ, USA, March 4-8, 2019*, pages 300–309. ACM, 2019.

[25] S. Pandey and G. Karypis. A self attentive model for knowledge tracing. 2019.

[26] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami,

L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 505–513, 2015.

[27] A. Pigeau, O. Aubert, and Y. Prié. Success prediction in moocs: A case study. 2019.

[28] A. Polyzou and G. Karypis. Feature extraction for classifying students based on their academic performance. In K. E. Boyer and M. Yudelson, editors, *Proceedings of the 11th International Conference on Educational Data Mining, EDM 2018, Buffalo, NY, USA, July 15-18, 2018*. International Educational Data Mining Society (IEDMS), 2018.

[29] C. Riener and D. Willingham. The myth of learning styles. *Change: The magazine of higher learning*, 42(5):32–35, 2010.

[30] C. Romero and S. Ventura. Data mining in education. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, 3(1):12–27, 2013.

[31] C. Romero and S. Ventura. Educational data mining and learning analytics: An updated survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, 10(3), 2020.

[32] A. Sheshadri, N. Gitinabard, C. F. Lynch, T. Barnes, and S. Heckman. Predicting student performance based on online study habits: A study of blended courses. *CoRR*, abs/1904.07331, 2019.

[33] S. Sonkar, A. S. Lan, A. E. Waters, P. Grimaldi, and R. G. Baraniuk. qdkt: Question-centric deep knowledge tracing. 2020.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[35] P. Wang, J. Xu, B. Xu, C. Liu, H. Zhang, F. Wang, and H. Hao. Semantic clustering and convolutional neural network for short text categorization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 352–357. The Association for Computer Linguistics, 2015.

[36] Y. Wang, N. Law, E. Hemberg, and U. O'Reilly. Using detailed access trajectories for learning behavior analysis. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge, LAK 2019, Tempe, AZ, USA, March 4-8, 2019*, pages 290–299. ACM, 2019.

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

APPENDIX

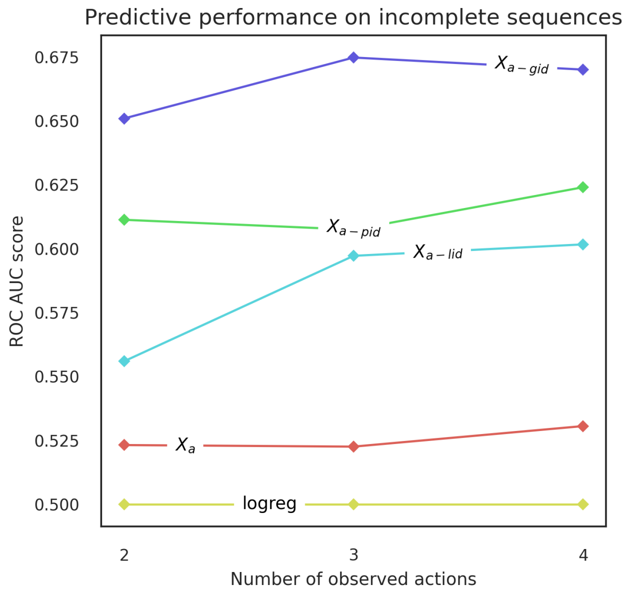


Figure 2: Comparison of validation ROC AUC scores of an LSTM on encoding schemes in the incomplete sequence scenario. We can see that percentage and local content id encoding schemes perform better than raw actions. As such, we would be able to detect whether a student is likely to fail from the first two actions already.

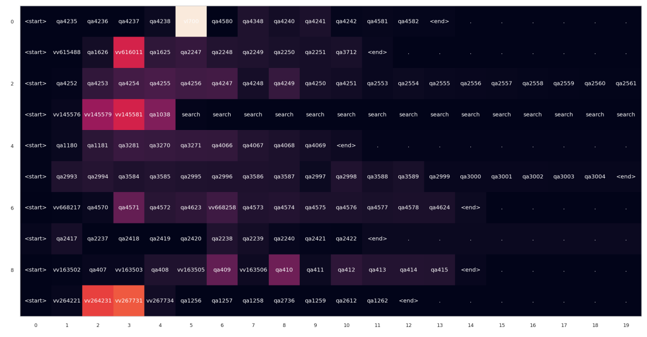


Figure 3: An attention heatmap of the RNN model with Bahdanau attention mechanism on action data (multiple sequences view). Higher scores (brighter colours) correspond to actions in the sequence that are more important for classifying the student as passing or not. If we use an encoding scheme which includes content id (such as the global content id here), we see which content items contribute more to the classification decision: for example, in the bottom row, the viewing of the video with id 264231 is more important for the network than the others.