# Early Prediction of Conceptual Understanding in Interactive Simulations

Jade Cock
EPFL
jade.cock@epfl.ch

Mirko Marras
EPFL
mirko.marras@epfl.ch

Christian Giang
EPFL
christian.giang@epfl.ch

Tanja Käser
EPFL
tanja.kaeser@epfl.ch

## ABSTRACT

Interactive simulations allow students to independently explore scientific phenomena and ideally infer the underlying principles through their exploration. Effectively using such environments is challenging for many students and therefore, adaptive guidance has the potential to improve student learning. Providing effective support is, however, also a challenge because it is not clear how effective inquiry in such environments looks like. Previous research in this area has mostly focused on grouping students with similar strategies or identifying learning strategies through sequence mining. In this paper, we investigate features and models for an early prediction of conceptual understanding based on clickstream data of students using an interactive Physics simulation. To this end, we measure students' conceptual understanding through a task they need to solve through their exploration. Then, we propose a novel pipeline to transform clickstream data into predictive features, using latent feature representations and interaction frequency vectors for different components of the environment. Our results on interaction data from 192 undergraduate students show that the proposed approach is able to detect struggling students early on.

## Keywords

skip-grams, early classification, interactive simulations, conceptual understanding

## 1. INTRODUCTION

Over the last years, interactive simulations have been increasingly used for science education (e.g, the PhET simulations alone are used over $45M$ times a year [1]). Interactive simulations allow students to engage in inquiry-based learning: they can design experiments, take measurements, and test their hypotheses. Ideally, students discover the principles and models of the underlying domain through their own exploration [2], but students often struggle to effectively learn in such environments [3, 4, 5]. A possible reason for this is that interactive simulations are usually complex and unstructured environments allowing students to choose their own action path [6]. Providing adaptive guidance to students has therefore the potential to improve learning outcomes.

Implementing effective support in interactive learning environments is a challenge in itself: the complexity of the environment makes it difficult to define a priori how successful student behaviour looks like. Previous research has focused on leveraging sequence mining and clustering techniques to identify the key features of successful interactions. For example, [7] have used an information theoretic sequence mining approach to detect differences in the interaction sequences of students with high and low prior knowledge, while [8] investigated the effects of prior knowledge activation. Other work [9] focused on detecting behaviours leading to the design of a correct causal explanation. [10] identified key factors for successful inquiry: focusing on an unknown component and building contrastive cases. Similarly, [11] found that the identification of the dependent variable and its isolated manipulations lead to a better quantitative understanding of the phenomena at hand. Another technique is to manually categorise students' log data and use the tags as ground truth for a classifier of successful inquiry behaviour [12]. [13] developed a dashboard displaying information about the mined sequences to guide teachers in building their lessons.

More work has focused on analysing and predicting students' strategies in different types of open ended learning environments (OELEs), such as educational games. Prior research in that domain has, for example, investigated students' problem solving behaviour [14], analysed the effect of scaffolding on students' motivation [15], extracted strategic moves from video learning games [16], detected different types of confusion [17], or identified students' exploration strategies [18].

Most of the previous work on OELEs has performed *a posteriori* analyses. However, in order to provide students with support in real-time, we need to be able to detect struggling students early on. Due to the lack of clearly defined student trajectories and underlying skills, building a model of students' learning in OELEs is challenging. A promising approach for early prediction in OELEs is the use of a clustering-classification framework [19]: in the (first) offline step, students are clustered based on their interaction data and the clustering solution is interpreted. The second step is online: students are assigned to clusters in real-time. This

Jade Cock, Mirko Marras, Christian Giang and Tanja Käser "Early Prediction of Conceptual Understanding in Interactive Simulations". 2021. In: Proceedings of The 14th International Conference on Educational Data Mining (EDM21). International Educational Data Mining Society, 161-171. https://educationaldatamining.org/edm2021/
*EDM '21* June 29 - July 02 2021, Paris, France

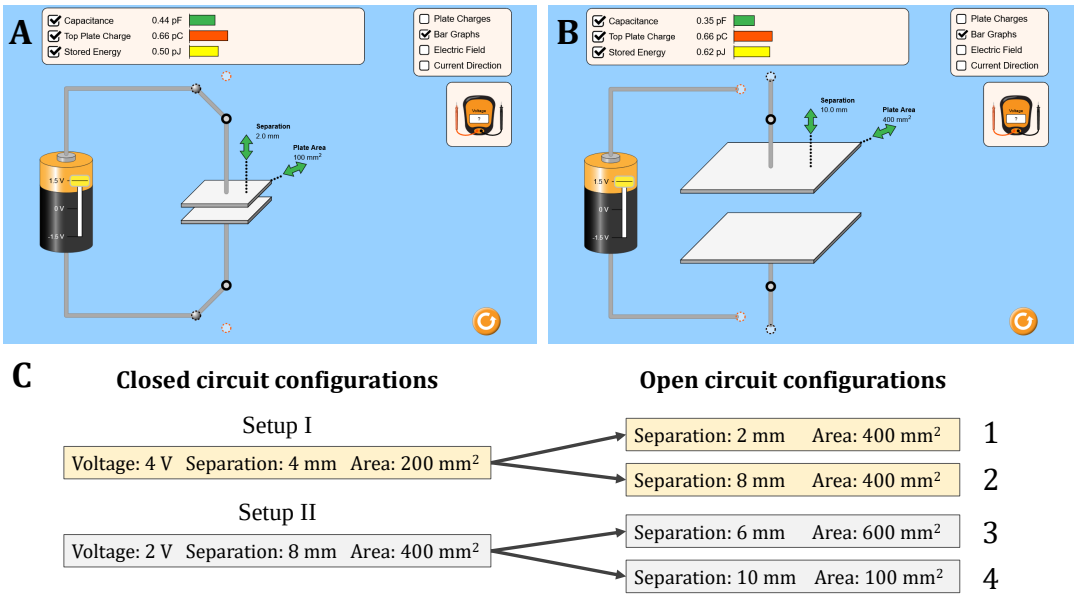


**Figure 1: User interface of the PhET Capacitor Lab with different plate parameters for a closed (A) and an open circuit (B). The two initial closed circuit configurations and the resulting four open circuit configurations presented to the participants in the capacitor ranking task (C). (Simulation image by PhET Interactive Simulations, University of Colorado Boulder, licensed under CC-BY 4.0, https://phet.colorado.edu).**

framework has been successfully applied to analyse and predict students' trajectories in mathematics learning [20], to differentiate between 'high' and 'low' learners [21], to build student models for interactive simulations [22], or to predict students' exploration strategies in an educational game [23].

In this paper, we aim at early predicting conceptual understanding based on students' log data from an interactive Physics simulation. All our analyses are based on data collected from 192 undergraduate Physics students interacting with a PhET simulation. We propose a novel pipeline for transforming clickstream data into predictive features using latent feature representations and frequency vectors. Then, we extensively evaluate and compare various combinations of predictive algorithms and features on different classification tasks. In contrast to previous work using unsupervised clustering to obtain student profiles [21, 20, 23, 22], our learning activity with the simulation includes a task specifically designed to assess students' conceptual understanding. With our analyses, we address three research questions: 1) Can students' interaction with the data be associated with the gained conceptual understanding? 2) Can conceptual understanding be inferred through sequence mining methods with embeddings? 3) Can the proposed methods be used for early predicting students' conceptual understanding based on partial sequences of interaction data?

Our results show that all tested models are able to predict students' conceptual knowledge with a high AUC when observing students' full sequences (offline). The best models are also able to detect struggling students early on and to provide a more fine-grained prediction of students' conceptual knowledge later during interaction.

## 2. CONTEXT AND DATA

All experiments and evaluations of this paper were conducted using data from students exploring an interactive simulation. In the following, we describe the learning activity, the data collection, and the categorisation of students' conceptual understanding at the end of the learning activity.

**Learning Activity.** The data for this work was collected in a user study where participants were asked to engage in an inquiry-based learning activity with the PhET Capacitor Lab simulation[1]. The Capacitor Lab is an interactive simulation with a simple and intuitive interface allowing users to explore the principles behind a plate capacitor (Fig. 1A and B). Specifically, students can load the capacitor by adjusting the battery and observe how the capacitance and the stored energy of the capacitor change when adjusting the voltage, the area of the capacitor plates or the distance between them. After loading the capacitor, the circuit can be opened through a switch, and students can again observe how manipulation of the different components influences capacitance and stored energy. Moreover, the simulation provides a voltmeter, while check boxes in the interface allow users to enable or disable visualisations of specific measures.

Based on this simulation, a learning activity was designed in which participants had to explore the relationships between the different components of the circuit and rank four different capacitor configurations by the amount of stored energy. The configurations were generated based on two initial setups (I and II, respectively) representing capacitors in a *closed circuit* with different settings for battery voltage,

[1]https://phet.colorado.edu/en/simulation/capacitor-lab-basics

plate area and separation (Fig. 1C). For each initial setup, two *open circuit* configurations were generated (i.e. configurations 1 & 2 from setup I and 3 & 4 from setup II) by opening the switch and then changing the values for plate area and separation. To complete this ranking task, participants were allowed to use the simulation for as much time as they needed. It should be noted that the values in the ranking task were chosen outside the ranges of the adjustable values in the simulation such that students could not simply reproduce the four configurations in the simulation, but had to solve the task by figuring out the relationships between the different components and the stored energy.

**Data Collection**. Data was collected from 214 first-year undergraduate Physics students who completed the capacitor ranking task as part of a homework assignment. While working with the simulation, students' interaction traces (i.e. clicks on check boxes, dragging of components, moving of sliders) were automatically logged by the environment. Moreover, it recorded students' final answers in the ranking task (i.e. the ranking of the four configurations). All data was collected in a completely anonymous way and the study was approved by the responsible institutional review board prior to the data collection (HREC number: 050-2020/05.08.2020). After a first screening of the data, several log files (9) were excluded because of inconsistencies in the data, and another 13 because they had barely any interaction (less than 10 clicks) with the environment. Removing these data points resulted in a data set of 192 students used for our analyses.

**Categorisation of conceptual understanding**. The design of the ranking task allows to relate students' responses to their conceptual understanding of a capacitor. For this purpose, we analysed the 16 (out of 24 possible) rankings submitted by the students with regards to conceptual understanding and grouped them accordingly. To this end, three concepts of understanding associated with the functioning of capacitors were evaluated in a top-down approach: answers were first separated by those representing an understanding of both the open and closed circuit (label *both*), and those only representing an understanding of the closed circuit (label *closed*). For those answers representing an understanding of both the open and closed circuit, two cases were distinguished. It was assumed that students who chose the only correct ranking of the configurations ("4213") gained an exhaustive understanding of the underlying concepts (label *correct*). Students who instead chose one of the other rankings were assumed to know how plate area and separation influence the stored energy in both the open and closed circuits, but failed to discover the influence of voltage on stored energy (label *areasep*). Within those answers that only represented an understanding of the capacitor's functioning in the closed circuit, we also distinguished between two cases. The first case represents the answer that would be considered correct if the task was to order the four configurations by capacitance instead of energy ("1324", label *capacitance*). Interestingly, 47 students (i.e. 24% of all students) submitted this ranking as an answer. The second case represents all other possible answers (label *other*) that could be submitted if (a part of) the closed circuit was understood.

Based on these three underlying concepts, we generated a decision tree with four leaves (each representing a group with similar conceptual understanding) and mapped all 16 rankings submitted by the students to the leaves (Fig. 2). These generated class labels will serve as ground truth labels for the classification task presented in the following sections.

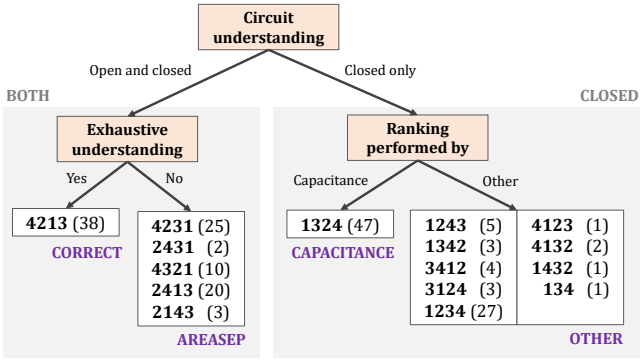


**Figure 2: Tree used to map the 16 different rankings submitted by the students to class labels associated with conceptual understanding of a capacitor. The different class labels are indicated in capitalised letters. The numbers in parentheses indicate the number of submissions for each ranking.**

## 3. METHOD

Using our proposed approach, we are interested in predicting the conceptual understanding students gain from interacting with the simulation. Therefore, we are solving a supervised classification problem, i.e. we aim at predicting the class labels (representing students' conceptual understanding) based on the observed student interactions. Our model building process to solve this classification problem consists of four steps (Fig. 3). We first extract the raw clickstream events from the logs and process them into action sequences. We then compute three different types of features for each action sequence and feed them into our classifiers.

**Event Logs**. From the simulation logs, we extract the clickstream data of each student $s$ as follows: anything between a mouse click/press and a mouse release qualifies as an *event*, while anything between a mouse release and mouse click/press is called *break*. Each *event* is then labelled by the component the user was interacting with at the mouse click, and chronologically arranged with the *breaks* into a sequence.

**Action Processing**. We distinguish three main components on the platform, whose values can be changed: 1) *the voltage*, 2) *the separation between the plates*, and 3) *the area of the plates*. An action on these components can be conducted in a) *an opened circuit* or b) *a closed circuit*, with the stored energy information display i) *on* or ii) *off*. We categorise each event involving these main actions by the combination of: the action on the component {1), 2), 3)}, the circuit state {a), b)}, and the stored energy display {i), ii)}. Any other event is categorised as 4) *other*.

The sequence of each student $s$ is now composed of chronologically ordered events (divided into the 13 different categories listed above) separated by breaks. The breaks may be caused by the student being inactive due to observing

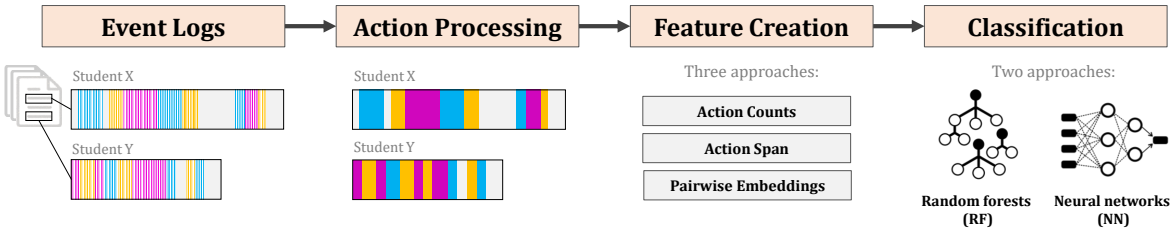


**Figure 3: Schematic overview of the classification pipeline. Raw clickstream events are extracted from log data and processed into an action sequence per user. Three types of features are computed for each action sequence. Classification is performed using Random Forests (RF) and Neural Networks (NN).**

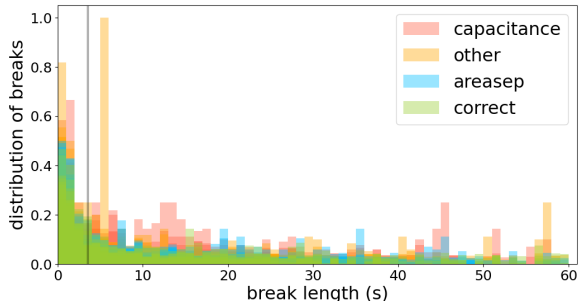


**Figure 4: Distribution of break lengths in seconds, across all students. The bold gray line denotes the 60% threshold.**

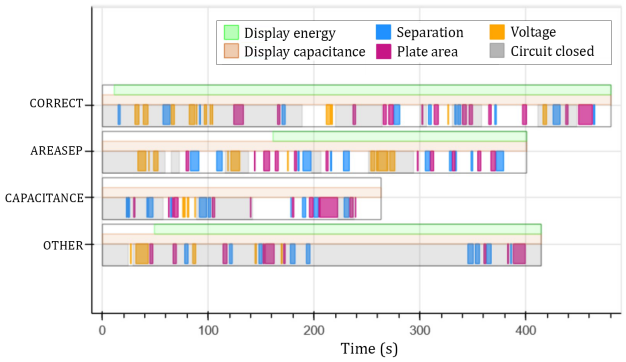


**Figure 5: Timeline of an exemplary student for each class label displaying the chronological sequence of interactions with the three main components of the simulation. The green and orange bars indicate whether the student displayed the capacitance and stored energy. The background indicates whether the interactions were conducted in a closed (grey) or open (white) circuit.**

the progression of a value, reflecting about an observation, or taking notes. Due to its definition as the period between a mouse release and a mouse click/press, a break may also appear due to logistic reasons, such as moving the mouse from one component in the simulation to another component. Indeed, the students' event sequences consist of many short breaks (Fig. 4). Like stop words in natural texts, our assumption is that these very short breaks, though very frequent in our sequences, do not contain much information. In fact, our classification over students' understanding may be impaired if those noisy states are not removed, like it is the case for sentiment analysis when stop words are not deleted [24]. To determine the threshold at which the breaks are removed, we plot the distribution of inactivity periods, and cut at the elbow of the curve for each student, which corresponds to a delimitation at 60%, i.e. for each student we keep the top 40% of breaks. We then categorise each of our remaining breaks similarly to our main action events: by component 5) *break*, circuit state {a), b)}, and stored energy display {i), ii)}, resulting in four different break categories.

The resulting sequence $\mathbf{r_s}$ for student $s$ is the chronological timeline of the student's events and breaks, divided into 17 categories. We refer to this timeline $\mathbf{r_s}$ as the *raw sequence* of *interactions* for the rest of the paper. We denote the length (corresponding to the total number of interactions of student $s$) of $\mathbf{r_s}$ with $N_s$, i.e. $|\mathbf{r_s}| = N_s$. On average, these sequences have a length of $N_s = 67.86 \pm 42.56$. In terms of seconds, the sequences $\mathbf{r_s}$ lasted on average $512.18 \pm 435.57$ seconds. We also introduce the notion of *time*, which we define related to a student's interactions: at time $t$ the length of the raw sequence of interactions for student $s$ is $t$, i.e. $|\mathbf{r_{s,t}}| = t$. We denote the maximum time of student $s$ with $T_s$, corresponding to the full sequence $\mathbf{r_s}$. Figure 5 visualises the timelines for an exemplary student of each class label. It can be observed that for these examples, certain aspects of conceptual understanding could be inferred by visual inspection (e.g. the *capacitance* student never activated the check box to visualise the stored energy). However, other differences in conceptual understanding are more difficult to detect by humans (e.g. the differences between the *correct* and *areasep* students).

**Feature Creation**. Next, we transform the interactions in each sequence to obtain three different types of features: *Action Counts*, *Action Span*, and *Pairwise Embeddings*.

To obtain the *Action Counts* features $F_{AC,s}$ for a student $s$, we first transform each interaction within the raw sequence $\mathbf{r_s}$ in a one-hot encoded vector, resulting in a 17-dimensional vector $\mathbf{h_{s,i}}$ for each interaction $i$ and hence, a sequence of

vectors $H_s = \{\mathbf{h_{s,i}}\}$ with $i = 1, ..., N_s$. To compute $\mathbf{f_{AC,s,t}}$ for student $s$ at time step $t$, we compute the average over $\mathbf{h_{s,i}}$ with $i = 1, ..., t$: $\mathbf{f_{AC,s,t}} = \frac{1}{t}\sum_{i=1}^{t} \mathbf{h_{s,i}}$. By using this aggregating technique, our features are translated to the (averaged) number of times each student has interacted in each of our categories. Therefore, for each student $s$, we end up with a feature set $F_{AC,s} = \{\mathbf{f_{AC,s,t}}\}$ with $t = 1, ..., T_s$.

The computation of the *Action Span* features $F_{AS,s}$ for a student $s$ is very similar. Rather than looking into the number of times a student $s$ has interacted with each of our components in a particular state, we look into the amount of time (in seconds) $s$ has spent in each of the categories. We first transform every interaction $i$ within $\mathbf{r_s}$ into a 17-dimensional vector $\mathbf{h_{s,i}}$: This vector is 0 for all dimensions but the dimension $d$ corresponding to the category the $i_h$ interaction belongs to. This representation is similar to a one-hot encoded vector, but instead of filling in a 1 at dimension $d$, we fill in the duration (in seconds) of interaction $i$. This results in a sequence of vectors $H_s = \{\mathbf{h_{s,i}}\}$ with $i = 1, ..., N_s$. We then compute the feature vector $\mathbf{f_{AS,s,t}}$ for student $s$ at time $t$ in two steps. In a first step, we again average over all vectors $\mathbf{h_{s,i}}$ up to time $t$, leading to $\tilde{\mathbf{f}}_{\mathbf{AS,s,t}} = \frac{1}{t}\sum_{i=1}^{t} \mathbf{h_{s,i}}$. We then normalise $\tilde{\mathbf{f}}_{\mathbf{AS,s,t}}$ to obtain $\mathbf{f_{AS,s,t}}$. By using this aggregation technique, our feature vector $\mathbf{f_{AS,s,t}}$ represents the relative amount of time student $s$ has spent in each category up to time $t$. For each student $s$, we end up with with a feature set $F_{AS,s} = \{\mathbf{f_{AS,s,t}}\}$ with $t = 1, ..., T_s$.

The third feature, *Pairwise Embeddings*, is fundamentally different from the the two other features: we replace each interaction in the raw sequence by an embedding vector which we obtain by training a pairwise skip-gram [25]. The architecture of such a network consists of two dense layers: an embedding layer followed by a classification layer. Usually applied to natural language applications (NLP), its primary goal is to predict the context of a word. Here, our pairwise skip-gram attempts to predict the behavior of a student in the simulation before and after performing a specific interaction. The skip-gram model can be formulated as:

$$\mathbf{p} = \text{softmax}(W_2 \cdot (W_1 \cdot a)) \tag{1}$$

It takes $a$, an interaction we wish to predict the context of as an input, and outputs $\mathbf{p}$, a probability vector which contains the likelihood of $a$ being surrounded by each possible *interaction*. $W_1$ and $W_2$ represent the weight matrices (embeddings). For each interaction $a$, we feed $2 \cdot w$ pairs into the network, where $w$ is the so-called window size of the model (context). The first element of each pair is $a$. The second element of each pair, the ground truth label, is one of the $w$ interactions preceding or following $a$. For example, a window size of $w = 2$ would yield the following pairs for action $a$: $(a, a_{-2}), (a, a_{-1}), (a, a_{+1}), (a, a_{+2})$.

In our case, to obtain the set of pairs $I_s$ for a student $s$, we first again transform each interaction within the raw sequence $\mathbf{r_s}$ in a one-hot encoded vector, resulting in a 17-dimensional vector $\mathbf{h_{s,i}}$ for each interaction $i$ and, hence, a sequence of vectors $H_s = \{\mathbf{h_{s,i}}\}$ with $i = 1, ..., N_s$. We then build the input pairs for each interaction $i$, i.e. $I_{i,s} = \{(\mathbf{h_{s,i}}, \mathbf{h_{s,j}})\}$ with $j \in \{-w, ..., w\} \setminus 0$. We obtain the set of pairs for all students as $I = \{I_s\}$ with $s = 1, .., S$, where $S$ is the total number of students.

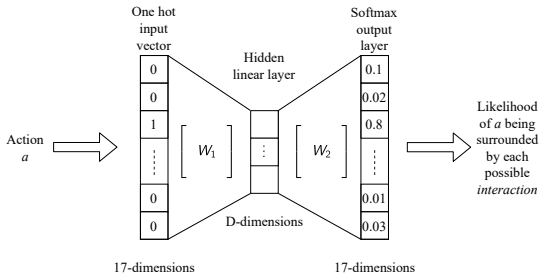


**Figure 6: Skip-gram architecture. After training of the skip-gram, the corresponding row of the weight matrix $W_1$ represents a structure preserving embedding of action $a$.**

After training the skip-gram model on $I$, we build the features $F_{PW,s}$ for each student $s$. The weighs $W_1$ of the hidden layer represent a structure preserving embedding of the interactions. In our case, $W_1$ has dimensions $17 \times D$ ($D$ is the embedding dimension). For each interaction $i$ of student $s$, we get the corresponding row $r$ in $W_1$, i.e. $\mathbf{r_{s,i}} = W_1 \cdot \mathbf{h_{s,i}}$. This again results in a sequence of vectors $R_s = \{\mathbf{r_{s,i}}\}$ with $i = 1, ..., N_s$. To compute $\mathbf{f_{PW,s,t}}$ for student $s$ at time step $t$, we compute the average over $\mathbf{r_{s,i}}$ with $i = 1, ..., t$: $\mathbf{f_{PW,s,t}} = \frac{1}{t}\sum_{i=1}^{t} \mathbf{r_{s,i}}$. Therefore, for each student $s$, we end up with a feature set $F_{PW,s} = \{\mathbf{f_{PW,s,t}}\}$ with $t = 1, ..., T_s$.

**Classification**. To perform the classification task, we explore two different approaches: *Random Forests* and *Fully Connected Deep Neural Networks*.

*Random Forests (RFs)* are simple, yet powerful machine learning algorithms. They consist of an ensemble of decision trees, each trained on a different subset of samples and a different subset of features. The decisions of each tree are then aggregated to determine the final prediction of a sample. The strength of this method is that overfitting is prevented through the randomisation of training samples and features during the training of each tree and that the strengths of several good classifiers are exploited. While RF classifiers are well tested and efficient to train, they require the input features to have the same dimension for every sample. We therefore train separate RF models for each time step $t$. The input features for the RF model for time step $t$ are $\{\mathbf{f_{M,s,t}}\}$, with $M \in AC, AS, PW$ and $s = 1, ..., S$, where $S$ denotes the number of students. The output of the RF is a vector $\mathbf{p_{RF,M,s,t}}$ of dimension $C$ (with $C$ denoting the number of classes) for each student $s$, which represents the probability of each class.

*Neural Networks (NNs)* were built with the idea of emulating neurons firing in our brain: their nodes are to the neurons what their edges are to their axons. The advantage of those deep networks is that they are able to model non-linear decision boundaries. However, the back propagation calculations make them relatively slow to train. In this work, we use a *Fully Connected Deep Neural Network* consisting of $d$ hidden dense layers and one classification layer with a softmax activation. Similar to RFs, our NN model requires features to have the same dimension for all the samples. We therefore also train the NN models for fixed points in time. The input features for the NN model for time step $t$ are

$\{\mathbf{f_{M,s,t}}\}$, with $M \in AC, AS, PW$ and $s = 1, ..., S$, where $S$ denotes the number of students. Due to the softmax activation, the output of the NN is a vector $\mathbf{p_{NN,M,s,t}}$ of dimension $C$ (with $C$ denoting the number of classes) for each student $s$, which represents the probability of each class.

# 4. EXPERIMENTAL EVALUATION

We evaluated the predictive performance of our classification pipeline on the collected data set (see Section 2). We conducted experiments to compare the performance of different model and feature combinations for the students' full data sequences as well as for early classification using partial sequences of the data, to answer our research questions.

**Experimental Setup**. We applied a train-test setting for all the experiments, i.e. parameters were fitted on a training data set and performance of the methods was evaluated on a test data set. Predictive performance was evaluated using the macro-averaged area under the ROC curve (AUC). We used the AUC as a performance measure as it is robust to class imbalance.

We performed all our experiments using different levels of detail for the classification task. As ground truth, we used the class labels presented in Fig. 2. Given the hierarchical nature of the decision tree separating the students in classes based on their conceptual knowledge, we performed the classification task focusing on three different levels of detail:

- *2-class case*: starting at the root of the decision tree (see Fig. 2), we divide students into two classes based on their understanding of the circuits: *both* (98 students) and *closed* (94 students).
- *3-class case*: going one step down in the hierarchy of the tree, we further divide the left branch of the tree (see Fig. 2) based on whether the students have completely understood all concepts (leading to a correct answer in our ranking task) or not. We therefore obtain three different classes: *correct* (38 students), *areasep* (60 students), and *closed* (94 students).
- *4-class case*: here, we also split the right branch of the tree (see Fig. 2) and divide the students into two groups based on whether they ranked the configurations in the task based on capacitance, resulting in four classes: *correct* (38 students), *areasep* (60 students), *capacitance* (47 students), and *other* (47 students).

For each of those three cases, we trained two types of classifiers (RF and NN) on our three different feature types (Action Counts - AC, Action Span - AS, Pairwise Embeddings - PW), using a stratified 10-fold **nested** cross validation. We kept the folds invariant across all experiments and stratified over the classes (according to the class labels of the 4-class case). Because of class imbalance, we used random oversampling for the training sets. We used a **nested** cross validation to avoid potential bias introduced by estimating model performance during hyperparameter tuning. This allowed us to tune the hyperparameters within the training folds (by further splitting them) and hold out the test sets for performance evaluation alone.

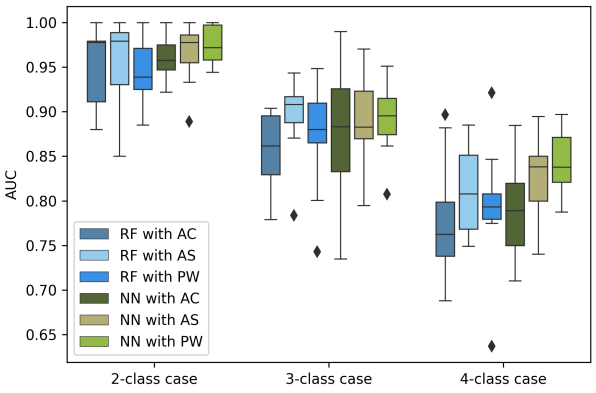


**Figure 7: AUC for 4-class, 3-class and 2-class cases using different model and feature combinations. Predictions are made at the end of the interaction with the simulation, i.e. based on the complete sequential interaction data of the students.**

For RF models, we tuned the following hyperparameters using a grid search: number of trees [5, 7, 9], number of features used at each decision level ['auto', 'all'], number of samples [bootstrap resampling of *training size* samples and balanced subsamples]. NN models were implemented using the *scikit-learn* library, trained for 300 epochs, and optimised for the log-loss function with the following hyperparameters: learning rate ['adaptive', 'invscaling'], initial learning rate [0.01, 0.001], solver ['adam', 'sgd'], hidden layer sizes and number [(32, 16), (64, 32), (64, 32, 16), (128, 64, 32, 16)], and activation function ['relu', 'tanh', 'identity'].

The skip-gram model providing our pairwise embedding features was implemented using the *TensorFlow* package. We trained the model for 150 epochs, with a window size of $w = 2$, a batch size of 16, and an embedding dimension of 15. We used categorical cross-entropy as the training loss. Because of its unsupervised nature, we trained the model on our whole dataset.

**Offline Classification**. In a first experiment, we were interested in assessing whether it is possible to associate students' behaviour in the simulation with their conceptual understanding achieved through the learning activity. This will be referred to as an offline classification task, since we are using students' complete interaction sequences $r_s$. The predictive performance in terms of AUC for the three classification problems (4-class case, 3-class case, and 2-class case) with a distinction between different model and feature combinations is illustrated in Fig. 7.

The results of this first experiment showed that for the 2-class case, all combinations of models and features reached very high average performances as quantified by their AUC scores (value range: $0.95 - 0.97$). The best mean score was achieved by the combination of NN with PW features ($AUC_{NN,PW} = 0.97$). However, it should be noted that the performance differences between the combinations were comparatively small. Using a one-way ANOVA, no statistically significant differences were found between the different groups ($F(5, 54) = 0.839, p = 0.528$). It seems that for this rather rough classification into groups of students who

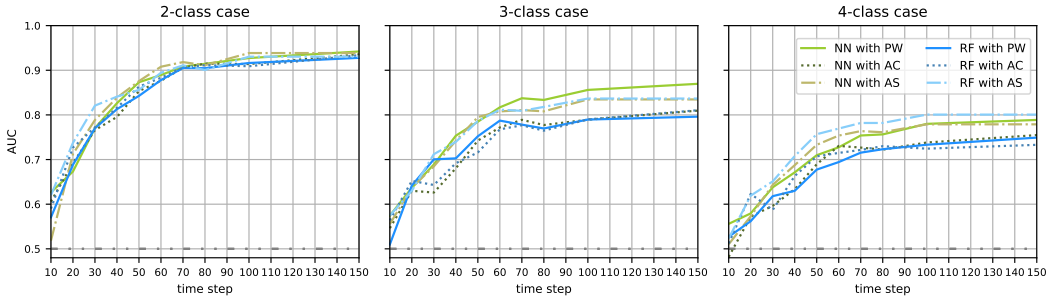


**Figure 8: AUC for 4 classes, 3 classes, and 2 classes using different model and feature combinations. Predictions are made over time, stopping at time step $t = 150$ as only very few students have longer interaction sequences.**

only understood the closed circuit and those who understood both the open and closed circuits, the different combinations of models and features perform equally well and with a high predictive accuracy.

By extending the classification task to the 3-class case, the AUC scores dropped in comparison to the 2-class case (value range: $0.86 - 0.90$). The lowest score was observed for RF with AC ($AUC_{RF,AC} = 0.86$), while best performances were observed for RF with AS ($AUC_{RF,AS} = 0.90$), NN with AS ($AUC_{NN,AS} = 0.89$), and NN with PW ($AUC_{NN,PW} = 0.89$). Similar to the 2-class case, no statistically significant differences were found between the different groups using a one-way ANOVA ($F(5, 54) = 0.740, p = 0.597$). This result illustrates that further dividing those who understood the functioning of the capacitor in both the open and closed circuit had a similar impact on the predictive performance of all model and feature combinations.

Finally, when evaluating the 4-class case, the most complex classification task, the mean AUC scores further dropped for all combinations (value range: $0.78 - 0.84$). The lowest performance was again observed for RF with AC ($AUC_{RF,AC} = 0.78$). Introducing the fourth class to the classification problem seemed to have a smaller negative impact on the AUC scores for NN with AS ($AUC_{NN,AS} = 0.83$) and NN with PW ($AUC_{NN,PW} = 0.84$), which obtain the best performances for the 4-class case. This observation was partially confirmed by a one-way ANOVA that showed a trend to statistical significance for differences between the combinations ($F(5, 54) = 1.989, p = 0.095$): as we increase the amount of classes, the p-value decreases.

The results of this first experiment show that it is possible to perform an offline prediction of students' conceptual understanding in the capacitor ranking task (see Section 2) based on the different combinations of models (NN or RF) and feature generation methods (AC, AS or PW) proposed in this work. From the entire data sequences of students' interactions with the simulation, we observed that predictive performance generally decreased when the complexity of the classification task was increased. While all combinations showed very good performances for the coarse classification of the 2-class case, AUC scores started to diverge more among combinations for the 3- and 4-class cases. Especially for the 4-class case, where differences became more visible with certain combinations showing a trend of better predictive performance (i.e. NN with PW and NN with AS) as compared to others (e.g., RF with AC).

**Predicting over Time**. In a second experiment, we were interested in assessing, whether we could predict students' conceptual knowledge for shorter interaction sequences, i.e. when not using students' full sequences, but only the first $t$ interactions. For all three classification cases (2-class, 3-class, and 4-class), we trained all model (RF, NN) and feature (AC, AS, PW) combinations for $t = 10, 20, ..., 150$ time steps. As described in Section 3, to compute the features $\mathbf{f_{AC,s,t}}$, $\mathbf{f_{AS,s,t}}$, and $\mathbf{f_{PW,s,t}}$ for a student $s$ at time step $t$, we only use the student's interactions $i$ up to that point in time, i.e. $i = 1, ..., t$. Similarly, at each time step $t$, the models are exclusively used to predict students whose interactions sequences contain a minimum of $t$ elements (i.e. $N_s \geq t$). For students with shorter interaction sequences (i.e. $N_s < t$), the last available prediction will be used. For example, for a student with 30 interactions ($N_s = 30$), we would make the first three predictions using the models for $t = 10$, $t = 20$, and $t = 30$ time steps. For the remaining time steps, the predictions from $t = 30$ will be carried over. We chose this approach for predicting because we assume that the student will leave the simulation after $N_s$ interactions and it therefore does not make sense to update the prediction afterwards. Figure 8 illustrates the predictive performance in terms of AUC for the different model and feature combinations and all classification cases.

As expected, for the 2-class case, all models achieve a high performance for long interaction sequences. The AUC of all NN models is larger than 0.9, starting at time step $t = 70$. Generally, the difference between the models and feature combinations for $t \geq 70$ is small. We also observe some model differences for earlier time steps, where the RF model with the action span features performs better than the other models. It achieves an AUC larger than 0.8 already at time step $t = 30$ ($AUC_{RF,AS} = 0.82$). Moreover, the NN model with action span is close to that performance at time step $t = 30$ ($AUC_{NN,AS} = 0.79$). Naturally, predicting at even earlier time steps is more difficult, but some of the models achieve a decent AUC of 0.7 already after observing 20 interactions with the simulation ($AUC_{RF,AS} = 0.74$, $AUC_{RF,AC} = 0.73$, $AUC_{NN,AS} = 0.71$).

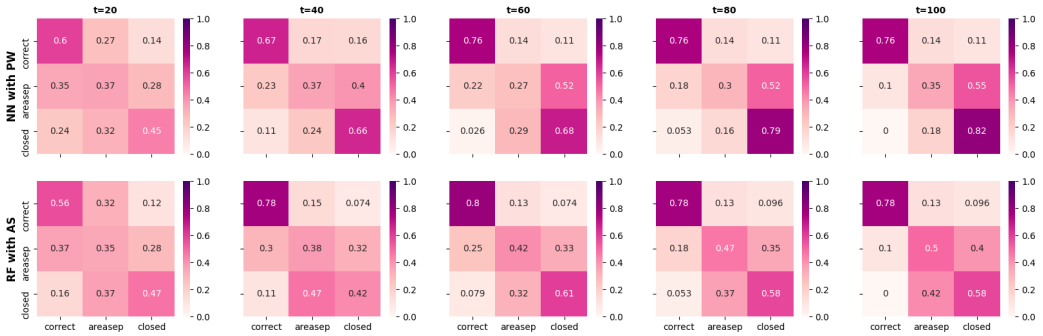


**Figure 9: Evolution of confusion matrices over time steps for the 3-class case: NN with PW (top) and RF with AS (bottom).**

For the 3-class case, the performances of all algorithms have decreased from time step $t = 10$ compared to the above problem. This is not surprising, as differentiating the correct students from the *areasep* students is not as straightforward as separating students who did not interact in the open circuit from the rest. Though, in the 2-class cases, all performances were close to one another, we notice that for the 3-class problem at time step $t = 40$, three model-features combinations take the lead ($AUC_{NN,PW} = 0.75$, $AUC_{RF,AS} = 0.74$, $AUC_{NN,AS} = 0.74$), while three fall behind ($AUC_{RF,PW} = 0.7$, $AUC_{RF,AC} = 0.69$ and $AUC_{NN,AC} = 0.68$) until $t = 70$, where the NN with PW outperforms all other model and feature combinations with an AUC of 0.87.

The variance across model-feature performances is larger in the 4-class case. From time step $t = 20$ already, RF with AS outperforms all other combinations, reaching an AUC of 0.8 at time step $t = 100$. Similarly to the 3-class problem, the same three models take the lead, while the others fall behind from time step $t = 70$ ($AUC_{NN,PW} = 0.75$, $AUC_{RF,AS} = 0.78$, $AUC_{NN,AS} = 0.76$ and $AUC_{RF,PW} = 0.72$, $AUC_{RF,AC} = 0.72$ and $AUC_{NN,AC} = 0.73$).

Given the fact the predictive performance in terms of AUC seems to be similar for the best performing models, we performed a more detailed analysis to assess how different the predictions of the several model and feature combinations were. In a real-world application (intervention setting), we would probably use a 2-class classifier to identify a coarse split (between classes *both* and *closed*) already at a relatively low number of time steps and use a 3-class method to provide a more detailed prediction later on. With the current model performance for the 4-class case, usage of a more detailed classifier seems not practicable. We therefore investigate the predictions of the two best models for the 3-class case: NN with PW and RF with AS. Figure 9 shows the confusion matrices for these two models for an increasing number of time steps. We do not show results for $t > 100$, as the predictive performance of the models does not improve much anymore for longer sequences (see also Fig. 8). While both models have a very similar predictive performance in terms of AUC up to time step 60, we can already see that the models evolve differently in terms of prediction. At time step $t = 40$, the RF model is already very accurate in detecting students from class *correct* ($\hat{p}(correct|c_{true} = correct) = 0.78$), while the NN model is less confident ($\hat{p}(correct|c_{true} = correct) = 0.67$). On the other hand, the NN model is already more accurate in identifying students from class *closed* ($\hat{p}(closed|c_{true} = closed) = 0.66$), while the RF model cannot identify students from this class well ($\hat{p}(closed|c_{true} = closed) = 0.42$). At time step $t = 60$, both models are almost equally accurate in identifiying students from class *correct* (NN: $\hat{p}(correct|c_{true} = correct) = 0.76$, RF: $\hat{p}(correct|c_{true} = correct) = 0.80$). The NN model is still better at classifying students from the class *closed*. Both models have trouble with correctly identifying students from class *areasep*. While the NN model tends to assign these students to class *closed* ($\hat{p}(areasep|c_{true} = closed) = 0.52$), the RF model is becoming better at correctly assigning them ($\hat{p}(closed|c_{true} = closed) = 0.42$). These observed trends continue to get stronger with an increasing number of time steps. At $t = 100$, the NN model is very accurate when it comes to classifying students from classes *correct* and *closed*. Students from class *areasep* have only a 35% chance of being correctly classified and a 55% chance of getting assigned to class *closed*. In practice, this would mean that 55% of the students would get more intervention (hints) than necessary. The RF classifier is also very accurate in detecting students from class *correct*, but is, however, not able to distinguish between students from class *closed* and class *areasep*. In practice, this would mean misclassified students from class *closed* would get less help than necessary and misclassified students from class *areasep* would get more help than necessary.

This experiment shows that we can (coarsely) classify students after observing a relatively low number of interactions. For the 2-class case, the AUC of the best model (RF with AS) is larger than 0.8 after $t = 30$ time steps. Naturally, the classification task is more complex for the 3-class and the 4-class cases. The best model on the 3-class case (NN with PW) achieves an AUC close to 0.8 at time step 50. The second analysis demonstrates that achieving a similar predictive performance in terms of AUC does not imply the same classification behaviour, i.e.the best models on the 3-class case (NN with PW and RF with AS) have different strengths. It has, however, one important limitation: students spend different amount of times on the simulation and therefore, the length of their interaction sequences varies. There are for example students with 80 interactions and other students with only 50 interactions. Performing the classification task at time step 40 is early for a student with

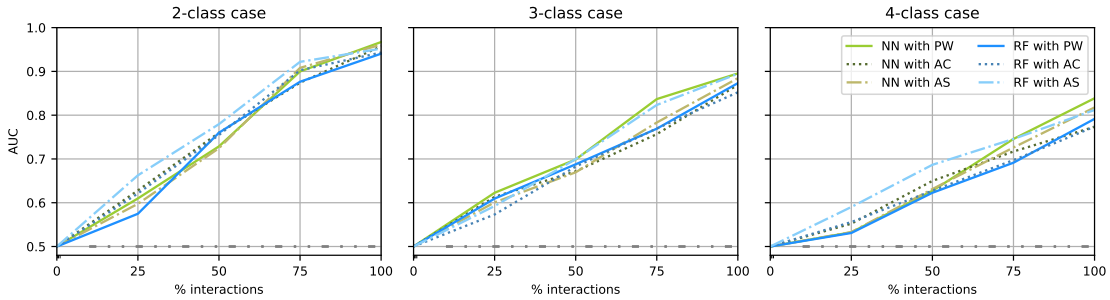


**Figure 10: AUC on the 2-class, 3-class and 4-class cases for different model and feature combinations. Predictions were made for 25%, 50%, 75% and 100% of the total number of interactions for each student.**

a total of 80 interactions. For a student with only 50 interactions in total, the prediction at time step 40 comes almost at the end of the interaction time with the simulation.

**Online (Early) Classification**. The third and last experiment addresses the limitations of the previous experiment. In this experiment, we were interested in assessing the "early" predictive performances of the different models using only a part of a student's sequence. Given the fact that the length of interaction sequences varies over the students, we did not align the sequences by absolute time steps, but by percentages of interactions. Specifically, we aimed at making the prediction for a student after having seen 25%, 50%, 75%, and 100% of the interaction sequence of this student. Note that this experiment does not require a re-training of our models. We just retrieve the predictions of the models for the corresponding time step $t$. For our example student with a total of 80 interactions, we retrieve the predictions of the models for time steps $t = 20$, $t = 40$, $t = 60$, and $t = 80$. Figure 10 shows the AUC of the models for all classification cases, with an increasing number of interactions (in %).

As expected, all model and feature combinations perform well for the 2-class case. To achieve a high classification accuracy, we do not need to observe the full sequence of a student. For all NN models, obtaining 75% of students' interactions is enough to achieve an AUC of around 0.9. With RF, the model using the action span features also obtains an AUC of more than 0.9 at 75% of the interactions ($AUC_{RF,AS} = 0.92$). The performance of the two other feature types is slightly lower ($AUC_{RF,AC} = 0.9$, $AUC_{RF,PW} = 0.88$). Naturally, predictive performance of all the models is lower when observing smaller parts of students' interaction sequences. If obtaining only the first 25% of students' interactions, there is more variation in the achieved AUC between models, with the best model (RF with AS) achieving an AUC of 0.66 and the worst model achieving an AUC of 0.57 (RF with PW). It is promising that the best model at 50% of the interactions exhibits an AUC of almost 0.8 ($AUC_{RF,AS} = 0.78$), which makes it a valuable candidate for a coarse early prediction and intervention, i.e. differentiating between students with a high conceptual understanding (class *both*) and students with a low conceptual understanding (class *closed*), early on.

Naturally, performance of the models for the 3-class case is overall lower as we are now differentiating the different levels of conceptual knowledge in a more fine-grained way. As we have seen in Fig. 9), it is difficult to differentiate between students from the left branch of the tree (i.e. *correct* vs. *areasep* in Fig. 2). Performance across models varies more for the 3-class case. When observing 75% of students interactions, the AUC of the worst model (NN with AS) amounts to 0.78, while the best model (NN with PW) has an AUC of 0.84. We also observe that the NN with PW features is consistently the best model, regardless of the amount of observed interactions, with an increasing gap to the other models. At 50% of interactions, the gap in performance among the three best models is still small ($AUC_{NN,PW} = 0.699$, $AUC_{RF,AS} = 0.7$, $AUC_{RF,PW} = 0.69$). It gets larger at 75% ($AUC_{NN,PW} = 0.84$, $AUC_{RF,AS} = 0.82$, $AUC_{NN,AS} = 0.78$). When observing the complete sequences of the students (i.e. 100% of the interactions), all models reach an AUC of 0.85 or higher (see also Fig.7).

Again, the performance decreases when moving to four classes, due to the increasing complexity (in terms of level of detail) of the classification task. The AUC of the best model (NN with PW) amounts to 0.83, while for the worst two models $AUC = 0.77$ (RF with AC, NN with AC). While all the models' AUC is lower than 0.7 when observing only 25% or 50% of students' interactions, interestingly there is a large gap between the best model (RF with AS) and all the other models (i.e. at 25%: $AUC_{RF,AS} = 0.59$, $AUC_{RF,AC} = 0.56$).

With this last experiment, we assessed the capabilities of our models to make predictions as early as possible during interaction with the simulation as a basis for intervention. By evaluating the models at different percentages of total interactions, we took into account the fact that the definition of 'early' depends on the student. Our results show that after observing the first 50% of interactions, we are able to reliably distinguish between students with a high and low conceptual understanding gained by the end of the learning activity (*both* and *closed*). At 75% of interactions, the best models are also able to provide a more fine-grained prediction (*correct*, *areasep*, and *closed*).

## 5. DISCUSSION

Over the last decade, interactive simulations of scientific phenomena have become increasingly popular. They allow students to learn the principles underlying a domain through

their own explorations. However, only few students possess the degree of inquiry skills and self-regulation necessary for effective learning in these environments. In this paper, we therefore explored approaches for an early identification of struggling students as a basis for adaptive guidance: we aimed at predicting students' conceptual knowledge while interacting with a Physics simulation. Specifically, we were interested in answering the following three research questions: 1) Can students' interaction with the data be associated with the gained conceptual understanding? 2) Can conceptual understanding be inferred through sequence mining methods with embeddings? 3) Can the proposed methods be used for early predicting students' conceptual understanding based on partial sequences of interaction data?

To answer the first research question, we analysed data from 192 first-year undergraduate Physics students who used an interactive capacitor simulation to solve a task in which they had to rank four capacitor configurations by their stored energy. Previous research has emphasised the importance of aligning instructional and assessment activities to implement pedagogically meaningful learning activities with educational technology [26, 27]. Since one objective of automatically detecting student learning behavior is to provide some kind of assessment (either formative or summative), the learning task presented in this work was designed to facilitate the application of sequence mining. The design of this learning task allowed us to relate all of the students answers to a certain level of conceptual understanding. Using a decision tree, we were able to map each ranking to one of the four labels representing groups of similar conceptual understanding. Our results show that all evaluated models were able to correctly associate students' sequential interaction data in the simulation with the generated labels, achieving high predictive performance when fed with full sequences (2-class case: $AUC > 0.9$, 3-class case: $AUC > 0.85$, 4-class case: $AUC > 0.75$). This high predictive power was also observed in [22], where they reached an accuracy of 85% when separating 'high' learners from 'low' learners based of full interaction sequences. However, despite their findings of a potential third cluster, they did not investigate the ternary classification task. In this paper, we increase the granularity of the labels in order to target more specific shortcomings in the knowledge of the students in order to provide them with more detailed feedback. We therefore conclude that we can answer research question 1) with yes.

The second research question investigates the benefits of latent features generated by skip-grams for offline classification tasks in the context of education. Usually applied to NLP problems, skip-grams have the ability to learn the context of a word in an unsupervised fashion. In our case, we use it to find the OELE behaviour of the students surrounding their interaction, and retrieve the embedding matrix of the neural network to create our latent representations. This approach has already been proven efficient to analyse student strategies in blended courses [28], but not for the identification of conceptual understanding. To evaluate the predictive power of latent feature representations, we trained two classifiers (NN and RF) on three types of feature (*Action Counts*, *Action Span* and *Pairwise Embeddings*) on the full sequences of students. At first, we notice that all model and feature combinations achieve a high AUC for all classification tasks (2-class case, 3-class case, and 4-class case). Though the ANOVA revealed no significant differences between the predictive performance of models with different types of features, we can observe that the NN with PW achieves a higher performance on average than all other combinations but the NN with AS. What is more, the performances in its first quartile dominate those from the third quartile of three model and feature combinations. Additionally, its performance variance is smaller than that of the NN with AS. This shows that pairwise embeddings generated by a skip-gram approach can be a valuable asset for finer-grained classification, even if no statistical difference was found with respect to the other model-feature combinations. We can therefore answer research question 2) with a partial yes.

To address the third research question, we assessed predictive performance of the proposed approach when only partial sequences of the students' interaction data were observed. We analysed the performances of our proposed approaches based on varying proportions of the available data and for classification tasks with different levels of complexity. The results of our experiments show that the proposed combinations of models and generated features allowed us to predict the correct class labels early on. The best models were able to reliably predict students' conceptual understanding for the 2-class case ($AUC \approx 0.8$) after having seen 50% of the students' interaction data. To reach a similar predictive performance for the more fine-grained 3-class and 4-class cases, the best models needed about 75% of the data. The findings from these experiments therefore represent a promising step towards early prediction of students' conceptual understanding in OELEs. We can therefore answer research question 3) with yes.

One of the limitations of this work is the unfeasibility to track whether students used external resources (other than the simulation) in order to rank the four capacitor configurations. This may bias the inference from the simulation usage to the extrapolated understanding level. Furthermore, due to our small sample size, we were able to only train shallow NN classifiers and skip-grams. Finally, the external validity of these experiments remains to be evaluated on other interactive simulations and different types of tasks.

To conclude, the proposed approach represents a promising step towards early prediction of students' learning strategies in interactive simulations, that moreover, can be associated with their level of conceptual understanding. The proposed learning activity seems to represent an interesting example for the design of learning tasks in OELEs that facilitates the association of detected student strategies with conceptual understanding through sequence mining. Future work could explore whether such designs could also be used to identify conceptual understanding at a more fine-grained level.

## 6. ACKNOWLEDGMENTS

We thank Kathy Perkins (PhET Interactive Simulations, University of Colorado Boulder) and Daniel Schwartz (Stanford University) for the valuable input on the study design and setup and for supporting data collection.

## References


[1] C. E. Wieman, W. K. Adams, and K. K. Perkins, "PhET: Simulations That Enhance Learning," *Science*, vol. 322, no. 5902, pp. 682–683, 2008.

[2] X. Fan and D. Geelan, "Enhancing students' scientific literacy in science education using interactive simulations: A critical literature review," *Journal of Computers in Mathematics and Science Teaching*, vol. 32, no. 2, pp. 125–171, 2013.

[3] R. E. Mayer, "Should there be a three-strikes rule against pure discovery learning? the case for guided methods of instruction," *American Psychologist*, vol. 59, no. 1, pp. 14–19, 2004.

[4] L. Alfieri, P. J. Brooks, N. J. Aldrich, and H. R. Tenenbaum, "Does discovery-based instruction enhance learning?" *Journal of Educational Psychology*, vol. 103, no. 1, pp. 1–18, 2011.

[5] P. A. Kirschner, S. J., and C. R.E., "Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching," *Educational Psychologist*, vol. 41, no. 2, pp. 75–86, 2006.

[6] I. Roll, V. Aleven, and K. R. Koedinger, "The invention lab: Using a hybrid of model tracing and constraint-based modeling to offer intelligent support in inquiry environments," in *Proc. ITS*, 2010, pp. 115–124.

[7] S. Perez, J. Massey-Allard, D. Butler, J. Ives, D. Bonn, N. Yee, and I. Roll, "Identifying productive inquiry in virtual labs using sequence mining," in *Proc. AIED*, 2017, pp. 287–298.

[8] C. Brand, J. Massey-Allard, S. Perez, N. Rummel, and I. Roll, "What inquiry with virtual labs can learn from productive failure: A theory-driven study of students' reflections," 2019, pp. 30–35.

[9] R. Baker, J. Clarke-Midura, and J. Ocumpaugh, "Towards general models of effective science inquiry in virtual performance assessments," *Journal of Computer Assisted Learning*, vol. 32, no. 3, pp. 267–280, 2016.

[10] E. Bumbacher, S. Salehi, M. Wierzchula, and P. Blikstein, "Learning Environments and Inquiry Behaviors in Science Inquiry Learning: How their Interplay Affects the Development of Conceptual Understanding in Physics," *Proc. EDM*, pp. 61–68, 2015.

[11] S. Perez, J. Massey-Allard, J. Ives, D. Butler, D. Bonn, J. Bale, and I. Roll, "Control of variables strategy across phases of inquiry in virtual labs," pp. 271–275, 2018.

[12] J. D. Gobert, M. S. Pedro, J. Raziuddin, and R. S. Baker, "From Log Files to Assessment Metrics: Measuring Students' Science Inquiry Skills Using Educational Data Mining," *Journal of the Learning Sciences*, vol. 22, no. 4, pp. 521–563, 2013.

[13] D. Tavares, K. Perkins, M. Kauzmann, and C. Velez, "Towards a teacher dashboard design for interactive simulations," in *Journal of Physics: Conference Series*, vol. 1287, no. 1, 2019, p. 012055.

[14] R. Sawyer, J. Rowe, R. Azevedo, and J. Lester, "Filtered time series analyses of student problem-solving behaviors in game-based learning." *Proc. EDM*, 2018.

[15] J. Sabourin, J. Rowe, B. Mott, and J. Lester, "Exploring affect and inquiry in open-ended game-based learning environments," in *Proc. ITS (Workshop on Emotions in Games for Learning)*, 2012, pp. 1–8.

[16] E. Rowe, R. Baker, J. Asbell-Clarke, E. Kasman, and W. Hawkins, "Building Automated Detectors of Gameplay Strategies to Measure Implicit Science Learning," in *Proc. EDM*, 2014, pp. 337–338.

[17] M. Eagle, E. Rowe, D. Hicks, R. Brown, T. Barnes, J. Asbell-Clarke, and T. Edwards, "Measuring Implicit Science Learning with Networks of Player-Game Interactions," in *Proc. CHI in Play*, 2015, pp. 499–504.

[18] T. Käser and D. Schwartz, "Modeling and analyzing inquiry strategies in open-ended learning environments," *IJAIED*, vol. 30, no. 3, pp. 504–535, 2020.

[19] S. Kardan and C. Conati, "A framework for capturing distinguishing user interaction behaviours in novel interfaces," *Proc. EDM,*, pp. 159–168, 2011.

[20] T. Käser, A. G. Busetto, B. Solenthaler, J. Kohn, M. von Aster, and M. Gross, "Cluster-Based Prediction of Mathematical Learning Patterns," in *Proc. AIED*, 2013, pp. 389–399.

[21] S. Amershi and C. Conati, "Combining Unsupervised and Supervised Classification to Build User Models for Exploratory Learning Environments," 2009, pp. 18–71.

[22] L. Fratamico, C. Conati, S. Kardan, and I. Roll, "Applying a framework for student modeling in exploratory learning environments: Comparing data representation granularity to handle environment complexity," *IJAEID*, vol. 27, no. 2, pp. 320–352, 2017.

[23] T. Käser and D. L. Schwartz, "Exploring Neural Network Models for the Classification of Students in Highly Interactive Environments," *Proc. EDM*, pp. 109–118, 2019.

[24] K. V. Ghag and K. Shah, "Comparative analysis of effect of stopwords removal on sentiment classification," in *Proc. IC4*. IEEE, 2015, pp. 1–6.

[25] Y. Goldberg and O. Levy, "Word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method," *CoRR*, vol. abs/1402.3722, 2014.

[26] T. Lauwers, "Aligning capabilities of interactive educational tools to learner goals," Ph.D. dissertation, CMU, Pittsburgh, PA, May 2010.

[27] C. Giang, "Towards the alignment of educational robotics learning systems with classroom activities," Ph.D. dissertation, EPFL, Lausanne, CH, 2020.

[28] N. Akpinar, A. Ramdas, and U. Acar, "Analyzing student strategies in blended courses using clickstream data," in *Proc. EDM*, 2020, pp. 6–17.