Online Academic Course Performance Prediction using Relational Graph Convolutional Neural Network

Hamid Karimi[1]*, Tyler Derr[1]*, Jiangtao Huang[2], Jiliang Tang[1]
[1] Michigan state University, {karimiha, derrtyle, tangjili}@msu.edu
[2] Nanning Normal University, China, hjt@gxtc.edu.cn

ABSTRACT

Online learning has attracted a large number of participants and is increasingly becoming very popular. However, the completion rates for online learning are notoriously low. Further, unlike traditional education systems, teachers, if any, are unable to comprehensively evaluate the learning gain of each student through the online learning platform. Hence, we need to have an effective framework for evaluating students' performance in online education systems and to predict their expected outcomes and associated early failures. To this end, we introduce Deep Online Performance Evaluation (DOPE), which first models the student course relations in an online system as a knowledge graph, then utilizes an advanced graph neural network to extract course and student embeddings, harnesses a recurrent neural network to encode the system's temporal student behavioral data, and ultimately predicts a student's performance in a given course. Comprehensive experiments on six online courses verify the effectiveness of DOPE across multiple settings against representative baseline methods. Furthermore, we perform ablation feature analysis on the student behavioral features to better understand the inner workings of DOPE. The code and data are available from https://github.com/hamidkarimi/dope.

Keywords

Online courses, Student behavior modeling, Knowledge graph, MOOC, Graph neural networks

1. INTRODUCTION

Online learning has higher dropout and failure rates than traditional education systems. For instance, the completion rates of Massive Open Online Courses (MOOCs), an extension of online learning technologies, are low (0.7%-52.1%, with a median value of 12.6%, reported by [20]). We also see similar situations in other online courses from universities such as Open University in the UK and China [19].

*Equal contribution and co-first author

Hamid Karimi, Tyler Derr, Jiangtao Huang and Jiliang Tang "Online Academic Course Performance Prediction using Relational Graph Convolutional Neural Network" In: *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)*, Anna N. Rafferty, Jacob Whitehill, Violetta Cavalli-Sforza, and Cristobal Romero (eds.) 2020, pp. 444 - 450

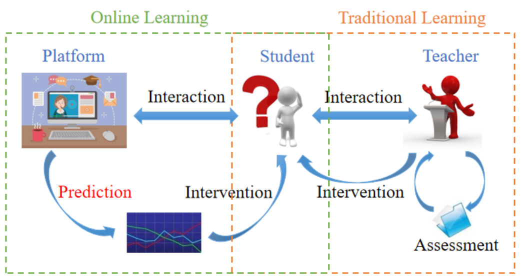


Figure 1: Visual comparison of the learning/intervention process between online and in-person education systems.

Furthermore, since students typically drop out early in the courses [33], the platform is desired to detect which student is likely to drop out (or fail) as early as possible to intervene and to hopefully prevent these negative outcomes. Then the question is how we can assess students' performance and detect those who are likely to drop out or fail in an online course. To answer this question, we first need to take a closer look at the online learning system and see how it differs from traditional learning.

As illustrated in Figure 1 (right side), in the traditional learning setting, instructors can interact with students, assess their performance, and take action to provide intervention if they sense a student is likely to perform poorly in the class. In online learning systems, however, the students primarily interact with the online platform, so we face a setting depicted in the left side of Figure 1. In this setting, there is inherently less interaction between students and instructors. More specifically, due to the high student-teacher ratio, teachers, if any, in the online learning systems are unable to comprehensively evaluate the learning gain of each student. Thus, we seek to develop a methodology that can harness the interactions of students with an online platform and accurately predict the course outcome (e.g., *pass* or *fail*). Such a system could then be used in real-time throughout the course to identify the students who are predicted to perform poorly and provide some intervention to them with the limited resources that are inherent in online systems.

Given the above discussion, we propose a framework named Deep Online Performance Evaluation (DOPE) to predict students' course performance in online learning. DOPE first models the student course relations of the online system as a knowledge graph. To incorporate an aggregated overview of the students and courses in the online system, DOPE learns student and course embeddings from our knowledge graph. More specifically, we employ a relational graph neural network [34] that can handle the rich attribute information found in our knowledge graph (e.g., student demographic data). Then, our proposed approach utilizes a recurrent neural network (RNN) to encode the temporal student behavioral data into some features. More specifically, the student behavioral data is coming from student click patterns extracted and aggregated into weekly snapshots that represent how they have interacted with the online learning system. Finally, the student and course embeddings (extracted from the knowledge graph) are combined with the encoded behavioral data extracted for the given student and course and are fed to a classifier to predict a student's performance. In summary, our contributions are as follows.

1. We propose the use of a knowledge graph to model complex online learning environments to allow more rich data to be extracted as compared to representing the data in a traditional unstructured way; and
2. Our proposed framework to predict student course outcomes contains two novel components, namely a relational graph neural network to extract student and course embeddings from the formed knowledge graph and a recurrent neural network model for encoding student behavioral data according to their clicks in the online system.

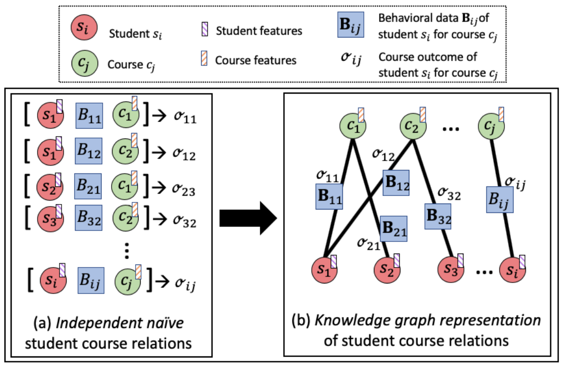
2. PROBLEM STATEMENT

Suppose from the set of courses in an online system we have a subset of m courses denoted as $\mathcal{C} = \{c_1, c_2, \cdots, c_m\}$. Furthermore, let there be n students having enrolled in at least one of the m courses in $\mathcal{C}$, which we denote as $\mathcal{S} = \{s_1, s_2, \cdots, s_n\}$. For each course c_j, we assume there are some course features that can be represented as the vector $\mathbf{f}_j \in \mathbb{R}^{d_c}$ with d_c being the dimension size after encoding the course features. Similarly for each of the students s_i we assume there has been some collected demographic information that can be represented as the vector $\mathbf{d}_i \in \mathbb{R}^{d_s}$ with d_s being the dimension size after having encoded the student demographic data. In addition to the demographic data, the system is assumed to have collected some sequential behavioral data for each student s_i enrolled in course c_j that we represent as $\mathbf{B}_{ij} = [\mathbf{B}^1_{ij}, \mathbf{B}^2_{ij}, \cdots, \mathbf{B}^k_{ij}]$ where $\mathbf{B}^w_{ij} \in \mathbb{R}^q$ represents an encoding of the behavior for student s_i during the w^{th} week of course c_j, k represents the number of weeks for which behavioral data was collected, and q is the dimension of the encoded weekly student behavior. In other words, we have a tensor of student behavioral data $\mathbf{B} \in \mathbb{R}^{n\times m\times k\times q}$. For each student s_i, we represent their performance outcome in course c_j as o_{ij}, where we assume there can be P outcomes (denoted by the set $p \in \mathcal{P}$).

Now, given the notations listed above, we seek to learn a model $f(.|\theta)$ having parameters θ such that it can predict the course student outcomes $\mathcal{O}$ as follows:

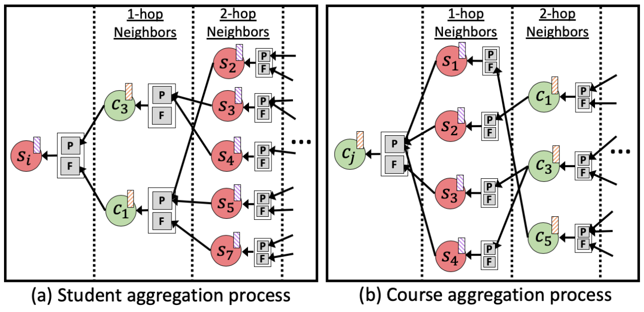
$$M(\mathcal{C}, \mathcal{S}, \mathcal{F}, \mathcal{D}, \mathbf{B}, \mathcal{O}, f(.|\theta)) \to \hat{\theta}$$

where we use M to denote the machine learning (artificial intelligence) process, $\mathbf{B}$ is used to represent the behavioral (e.g., click) data for a given set of courses $\mathcal{C}$ using only the first k weeks of data, $\mathcal{F}$ represents the set of course features of $\mathcal{C}$, $\mathcal{D}$ denotes the set of demographic data for the students



Student s_i
Course c_j
Student features
Course features
Behavioral data $\mathbf{B}_{ij}$ of student s_i for course c_j
Course outcome of student s_i for course c_j
(a) *Independent naïve* student course relations
(b) *Knowledge graph representation* of student course relations

Figure 2: Visualizing the traditional representation used in prior supervised learning prediction models as compared to our knowledge graph representation.



1-hop Neighbors
2-hop Neighbors
(a) Student aggregation process
(b) Course aggregation process

Figure 3: Visualizing the aggregation process in how both a student and course embedding are formed from their knowledge graph multi-hop neighborhood.

in $\mathcal{S}$, $\mathcal{O}$ represents the performance outcomes of the students in $\mathcal{S}$ and the learned parameters of $f(.|\theta)$ are given by $\hat{\theta}$.

3. PROPOSED MODEL

In this section, we explain our proposed model in detail.

3.1 Knowledge Graph Representation

We first model the historical online course data in the form of a knowledge graph, as shown in Figure 2. Our knowledge graph formulation in Figure 2(b) offers a richer representation than a traditional independent naive student course relation representation shown in Figure 2(a). This is because through this graph structure we can leverage the relations between students and courses beyond that seen in Figure 2(a). We let $\mathcal{G} = \{\mathcal{C}, \mathcal{S}, \mathbf{X_c}, \mathbf{X_s}, \mathbf{B}, \mathbf{A}\}$ represent a knowledge graph $\mathcal{G}$ containing the set of m course nodes $\mathcal{C}$, set of n student nodes $\mathcal{S}$, course features $\mathbf{X_c} \in \mathbb{R}^{n\times d_c}$ constructed from $\mathcal{F}$, student demographic features $\mathbf{X_s} \in \mathbb{R}^{m\times d_s}$ constructed from $\mathcal{D}$, the behavioral data $\mathbf{B}$ representing complex sequential edge features, and an adjacency tensor $\mathbf{A} \in \mathbb{R}^{n\times m\times P}$ constructed from the P different student-course outcome relations where $\mathbf{A}^p_{ij} = 1$ if $o_{ij} \in \mathcal{O}$ and $o_{ij} = p$ (with $\mathbf{A}^p_{ij} = 0$ otherwise). Now, given the knowledge graph G, we seek to extract student and course embeddings by using a relational graph neural network.

3.2 Relational Graph Neural Network

Recently, graph neural networks (GNNs) [38, 39] have become increasingly popular due to their ability to utilize deep learning on graph structure data. One popular class of GNNs is the graph convolutional networks (GCNs) [5, 27, 8, 7], which are constructed with roots from the classical CNNs. The general idea of these GCN models is that we would like to learn a better set of latent features. In the context of our problem, to better understand and represent a student, rather than directly using their features alone, we could use a 1-layer GCN that would incorporate the features of all the courses that the student has taken. For example, in Figure 3(a), the 1-hop neighbors would be utilized in a 1-layer GCN model taking into consideration the course c_3 that they passed and c_1 that they failed. Then, it is natural to see in Figure 3(a) that using a 2-layer GCN would furthermore incorporate the 2-hop neighbors which would include information from all the classmates of s_i for each of the two courses they have taken, and thus providing further context into learning a more comprehensive embedding for student s_i. We specifically harness the ability of a relational graph convolutional network [34]. Next we will provide the details on how the first layer (or equivalently a 1-layer) GCN is able to construct learned representations ${\mathbf{h_s}}_i^1$ and ${\mathbf{h_c}}_j^1$ for the student s_i and course c_j, respectively, from the initial student features $\mathbf{X_s}$, course features $\mathbf{X_c}$, and adjacency tensor $\mathbf{A}$ in our knowledge graph representation.

–**First Layer Embeddings.** First, we recall that connections between students and courses are stored in the tensor $\mathbf{A}$ where $\mathbf{A}_{ij}^p = 1$ if $o_{ij} \in \mathcal{O}$ and $o_{ij} = p$ (with $\mathbf{A}_{ij}^p = 0$ otherwise). Thus, we define for a student s_i their set of courses for which they had outcome p as $\mathcal{N}_s^p(s_i)$. Similarly, we define for a course c_j their set of students that received the outcome p as $\mathcal{N}_c^p(c_j)$. Now, given these new notations, we can define the first layer representations ${\mathbf{h_s}}_i^1$ and ${\mathbf{h_c}}_j^1$ for the student s_i and course c_j, respectively, as follows:

$$ {\mathbf{h_s}}_i^1 = \sigma\Big(\mathbf{W}_{self}^1 \mathbf{X_s}_{[i]} + \sum_{p \in \mathcal{P}} \frac{1}{|\mathcal{N}_s^p(s_i)|} \sum_{c_j \in \mathcal{N}_s^p(s_i)} \mathbf{W}_p^1 \mathbf{X_c}_{[j]}\Big) \quad (1) $$

$$ {\mathbf{h_c}}_j^1 = \sigma\Big(\mathbf{W}_{self}^1 \mathbf{X_c}_{[j]} + \sum_{p \in \mathcal{P}} \frac{1}{|\mathcal{N}_c^p(c_j)|} \sum_{s_i \in \mathcal{N}_c^p(c_j)} \mathbf{W}_p^1 \mathbf{X_s}_{[i]}\Big) \quad (2) $$

where σ is an element-wise non-linear activation function (e.g., $\mathrm{ReLU}(\cdot) = \max(0, \cdot)$ [13]), $\mathbf{X_s}_{[i]}$ denotes the student features for s_i, $\mathbf{X_c}_{[j]}$ denotes the course features for c_j, $\mathbf{W}_{self}^1$ is used to transform the self features from the original features, and $\mathbf{W}_p^1$ is used for transforming the features that are linked through the relation (i.e., course outcome type) p for the first layer.

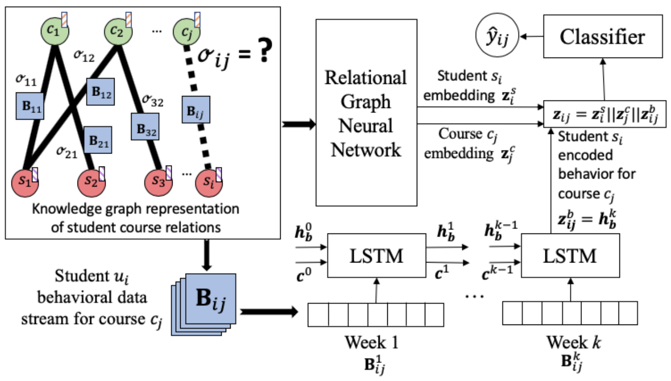
–**Final Student Embeddings.** If we assume having L layers in our GCN model, we can then first define the last layer where we will obtain the student embedding $\mathbf{z}_i^s = {\mathbf{h_s}}_i^L$ for s_i as follows:

$$ \mathbf{z}_i^s = \sigma\left(\mathbf{W}_{self}^L {\mathbf{h_s}}_i^{L-1} + \sum_{p \in \mathcal{P}} \frac{1}{|\mathcal{N}_s^p(s_i)|} \sum_{c_j \in \mathcal{N}_s^p(s_i)} \mathbf{W}_p^L {\mathbf{h_c}}_j^{L-1}\right) $$

where ${\mathbf{h_s}}_i^l$ represents the representation of student s_i at layer l of the GCN. Note that if we were to use a 2-layer GCN (i.e., $L = 2$) then ${\mathbf{h_s}}_i^{L-1} = {\mathbf{h_s}}_i^1$ would be coming from Eq. (1) and similarly ${\mathbf{h_c}}_j^{L-1} = {\mathbf{h_c}}_j^1$ from Eq. (2).

–**Final Course Embeddings.** If we assume having L layers in our GCN model, we can then first define the last layer where we will obtain the course embedding $\mathbf{z}_j^c = {\mathbf{h_c}}_j^L$ for c_j as follows:

$$ \mathbf{z}_j^c = \sigma\left(\mathbf{W}_{self}^L {\mathbf{h_c}}_j^{L-1} + \sum_{p \in \mathcal{P}} \frac{1}{|\mathcal{N}_c^p(c_j)|} \sum_{s_i \in \mathcal{N}_c^p(c_j)} \mathbf{W}_p^L {\mathbf{h_s}}_i^{L-1}\right) $$



c_1
c_2
...
c_j
σ_{11}
σ_{12}
σ_{ij} = ?
$\mathbf{B}_{11}$
$\mathbf{B}_{12}$
σ_{32}
$\mathbf{B}_{ij}$
σ_{21}
$\mathbf{B}_{21}$
$\mathbf{B}_{32}$
s_1
s_2
s_3
...
s_i
Knowledge graph representation of student course relations
Relational Graph Neural Network
Student s_i embedding $\mathbf{z}_i^s$
Course c_j embedding $\mathbf{z}_j^c$
$\hat{y}_{ij}$
Classifier
$\mathbf{z}_{ij} = \mathbf{z}_i^s||\mathbf{z}_j^c||\mathbf{z}_{ij}^b$
Student s_i encoded behavior for course c_j
$\mathbf{z}_{ij}^b = \boldsymbol{h}_b^k$
$\boldsymbol{h}_b^0$
$\boldsymbol{c}^0$
LSTM
$\boldsymbol{h}_b^1$
$\boldsymbol{c}^1$
$\boldsymbol{h}_b^{k-1}$
$\boldsymbol{c}^{k-1}$
LSTM
...
Student u_i behavioral data stream for course c_j
$\mathbf{B}_{ij}$
Week 1 $\mathbf{B}_{ij}^1$
Week k $\mathbf{B}_{ij}^k$

Figure 4: Visualizing the entire DOPE model consisting of both the relational graph neural network and recurrent neural network components.

where ${\mathbf{h_c}}_j^l$ is the embedding of student c_j at layer l of the GCN. Next, we will discuss how DOPE uses an RNN to encode a student's sequential behavior data associated with a given course.

3.3 Encoding Student Behavioral Data

In this part, we discuss how to encode the sequential edge features i.e., behavioural student data. To recall, when a student s_i is currently enrolled in a course c_j, by the k^{th} week we will have the features $\mathbf{B}_{ij} = [\mathbf{B}_{ij}^1, \mathbf{B}_{ij}^2, \cdots, \mathbf{B}_{ij}^k]$. To better represent the behavioral data, we utilize a Long-Short Term Memory (LSTM) [16], which is an effective RNN variant that has been designed to extract temporal features from sequential data e.g., videos [25] , speech [14, 25], and text [24, 23]. Furthermore, it has shown great abilities to capture temporal online user behaviors [26]. We fix the length of the behavior feature sequence for all students to be k (e.g., 10 weeks). Then for a given behavioral sequential data $\mathbf{B}_{ij}$, at each week $t \in [1, k]$, an LSTM unit takes the t-th week's click feature vector $\mathbf{B}_{ij}^t$ as the input and uses LSTM formulation [16] to produce the output behavioral vector $\mathbf{h}_b^t$. The final output of the LSTM is $\mathbf{h}_b^k \in \mathbb{R}^{e^b}$ (i.e., output of last LSTM unit) when given the sequence $\mathbf{B}_{ij}$ as input. Then, we set the encoded behavior of student s_i for course c_j as the e^b dimensional vector $\mathbf{z}_{ij}^b = \mathbf{h}_b^k$.

3.4 Final Course Performance Classifier

Here we combine student and course embeddings from the relational graph convolutional as well as encoded behavioral data and feed into a classifier. This can be seen in Figure 4. Given the student embedding $\mathbf{z}_i^s$ for student s_i, course embedding $\mathbf{z}_j^c$ for course c_j, encoded student behavior of s_i in the course c_j as $\mathbf{z}_{ij}^b$ we form the final feature representation as follows:

$$ \mathbf{z}_{ij} = \mathbf{z}_i^s||\mathbf{z}_j^c||\mathbf{z}_{ij}^b $$

where $||$ denotes concatenation and we concatenate the three components together into a single $(e^s + e^c + e^b)$ dimensional representation. For training DOPE, we use supervised learning such that labels are the outcome performances from the historical data $o_{ij} \in \mathcal{O}$ and matched with the training student and course pair (s_i, c_j). More specifically, we construct a minibatch set $\mathcal{M}$ that contains triplets of the form (s_i, c_j, T) where $T = o_{ij}$ (i.e., the course outcome type) and we assume the outcome type set $\mathcal{T}$ where $|\mathcal{T}| = p$ since there are p course outcome types. The objective is then formalized in the following:

$$ \mathcal{L} = \frac{1}{|\mathcal{M}|} \sum_{(s_i, c_j, T = o_{ij}) \in \mathcal{M}} \log \frac{\exp(\theta_T^{MLG} \mathbf{z}_{ij})}{\sum\limits_{T' \in \mathcal{T}} \exp(\theta_{T'}^{MLG} \mathbf{z}_{ij})} \quad (3) $$
$$ + \lambda Reg(\theta^{RGCN}, \theta^{LSTM}, \theta^{MLG}) $$

where the classifier first maps $\mathbf{z}_{ij}$ to a p dimensional vector through the parameters θ^{MLG} (since we have p different outcomes, i.e.,

Table 1: The description of the dataset.

Name	Train Periods	Test Periods	#Students
SS_1	2013J	2014J	735
SS_2	2013B, 2013J	2014B, 2014J	6622
SS_3	2013J, 2014B	2014J	2366
ST_1	2013B, 2013J	2014B, 2014J	5745
ST_2	2013J, 2014B	2014J	2685
ST_3	2013B, 2013J	2014B, 2014J	7092

link labels in the knowledge graph) and then utilizes the softmax function to get the outcome probabilities.

4. EXPERIMENTS

In this section, we conduct some experiments to verify the working of our proposed method.

4.1 Dataset and Experimental Settings

Online education platforms utilize virtual learning environments (VLEs) to collect records about all students' interactions and provide the opportunity for analyzing students' learning behavior. In this study, we use the data of The Open University Learning Analytics Dataset (OULAD) [29], which contains 22 open university courses for years 2013 and 2014 and 32,593 students. The dataset includes student demographic information, student assessment results, and daily interactions with the university's VLEs (10,655,280 entries). For each year, courses are offered in two distinct modules denoted as B and J (essentially they are similar to 'semester' in the conventional education system) where each module takes around 35 to 40 weeks long. The outcome of a course for a student can have four different categories including *Distinction*, *Pass*, *Fail*, and *Withdrawn*. We use OULAD and select three social science courses (i.e., SS1, SS2, and SS3) and three Science, Technology, Engineering, and Mathematics (STEM) courses (i.e., ST1, ST2, and ST3) as demonstrated in Table 1.

To represent the behavioral data, we count the different number of weekly clicks a student makes e.g., accessing resources, webpage click, forum click, quiz attempt, and so on. The size of each weekly behavioral vector is 20. Further, course attributes include two one-hot encoding vectors, one for representing a course among 6 courses, and the other one for holding either the course is social science or STEM. Train and test periods are shown in Table 1. We use 10% of the training data as a validation set to tune the hyper-parameters. The implementation is done using PyTorch package [30]. Each simulation is run for 200 epochs with a learning rate set to 0.001 and a decaying rate of 0.99 every 100 steps. As for the evaluation metric, we use weighted F1 score which is the harmonic mean of recall and precision.

4.2 Baseline Methods

We compare the performance of DOPE with the following baseline methods.

- **SVM**. In this baseline method, we concatenate the course attributes and students' demographic features as well as weekly click data (i.e., behavioral data) into a single vector and feed it to a support vector machine with radial basis function kernel.
- **LR**. This is similar to SVM except we use logistic regression for classification. The reason for including this baseline is to measure the online course performance prediction problem using a simple classification method without any kernel or non-linearity.
- **$\mathbf{DOPE_{FCN}}$**. This is a variation of DOPE where instead of modeling behavioral data with an LSTM, we use a fully connected network. The reason for including this method is to evaluate the effectiveness of the way we model sequential behavioral data.

We compare DOPE with the baseline methods for the different numbers of weekly click data i.e., 5, 10, 15, and 20 weeks. By doing so, we can measure how effective DOPE is in the early prediction of a student's course performance prediction. We note that 20 weeks is almost half of a course period when there is still adequate time for intervention in the case of prediction as failure.

4.3 Binary Classification

As mentioned begore, our dataset includes 4 distinct labels for a student's performance in a course, namely *Distinction*, *Pass*, *Fail*, and *Withdrawn*. In this section, we merge *Distinction* and *Pass* into a single class "Pass" and *Fail* and *Withdrawn* into a single class "Fail" and then perform a binary classification. Figure 5 illustrates the experimental results for all courses. We make the following observations based on the results presented in Figure 5.

- In general, the more weekly click data is introduced, the better we can predict the students' outcomes. DOPE enjoys more of such performance increase as compared to other methods. In particular, as early as 20 weeks from the start of a course (i.e., almost in the middle of a course duration), it can predict student's outcomes with very high performance. This allows teachers or online course administration to take actionable and interventive measures to help students with poor performance.
- DOPE achieves a better performance than $\mathrm{DOPE_{FCN}}$. This shows the fact the LSTM component as a machinery extracting temporal features from click behaviors is necessary and affects the model's predictive power.
- DOPE is shown to be effective for all courses as we can observe it achieves an F1 score of more than 0.8 across all courses when 20 weeks of click data are considered.

4.4 4-class Classification

In this part, we compare the performance of DOPE with baseline methods for a 4-class classification setting whose experimental results are demonstrated in Figure 6. We make the following observations based on the results in Figure 6.

- The observations we made for binary classification hold for 4-class classifications as well. In particular, DOPE still outperforms baseline approaches, more weekly click data is helpful in course outcome prediction, and the LSTM can effectively handle sequential that than simple concatenation followed by a fully connected network model (i.e., $\mathrm{DOPE_{FCN}}$).
- Since more classes are considered, compared to binary classification, the 4-class classification is a harder task. In particular, now *Withdrawn* is considered as a separate class, which might be "conceptually" hard for a model to discern from *Fail*.

4.5 Behavioral Feature Analysis

Since behavioral data (i.e., click data) plays an essential role in determining a student's performance, we conduct a feature analysis experiment investigating the importance of each behavior type. A similar feature analysis has been performed to discover great insights into human behaviors [21]. To this end, we follow an ablation feature analysis where at each time we include one feature type and suppress the rest (setting their values to zero) and then acquire the F1 score from the model. We do this experiment for the binary classification and the case when 20 weeks of click data is included. Figure 7 demonstrates the results and we make the following observations accordingly.

- For all courses, feature type *homepage* is associated with a high F1 score. This seems reasonable since most of the click activity occurs on the main page of the platform interface.
- Interestingly, clicks and activities in *forums* have an influential role in predicting fail or pass of a student in a course. This is in line with previous [15, 31] where they showed that MOOC forum activities correlate with a student's academic performance.

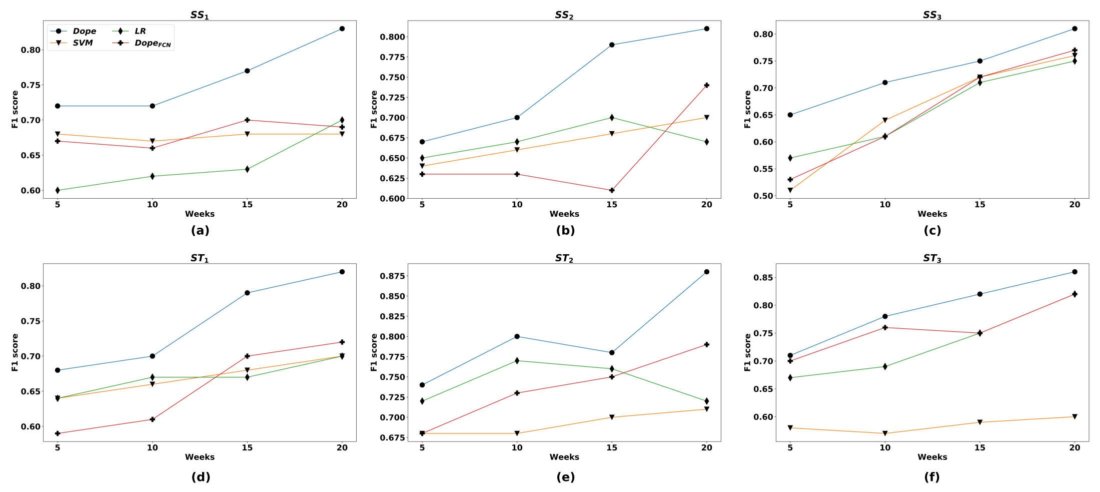


Figure 5: Comparison results for binary classification using four different amounts of included weekly click data.

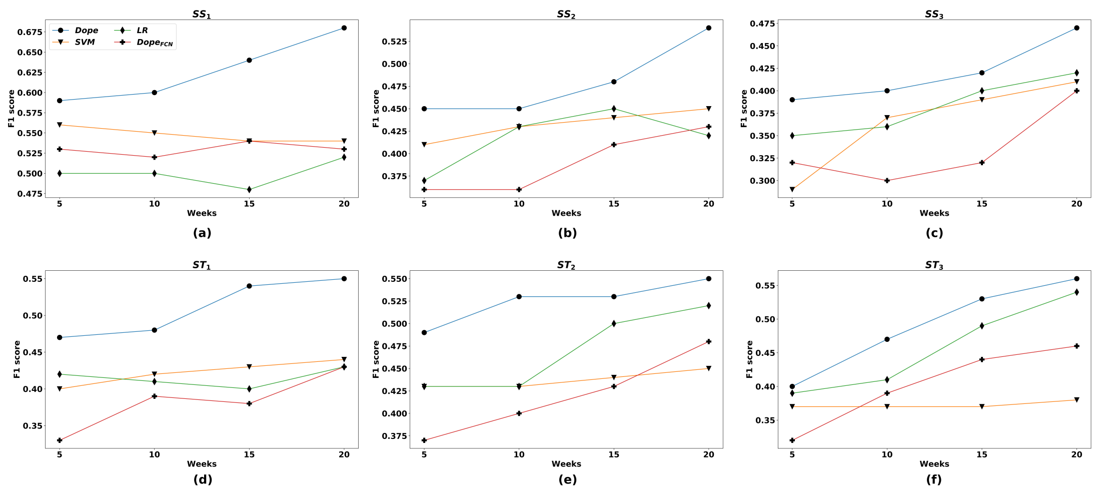


Figure 6: Comparison results for 4-class classification using four different amounts of included weekly click data.

- Unique behavioral importance profile of a course can aid policymakers, administrators and even course interface designers to prepare the course materials in a more informed way. For instance, we can observe that attribute *wiki* is playing an important role in performance prediction of ST_2 while its effect is negligible for other courses. This can be indicative of materials of the course ST_2 to be requiring more wiki access and consequently, the content can be changed accordingly.

Based on the observations above, we can conclude that DOPE encodes behavioral data in an intuitive manner that conforms to previous studies' findings as well.

4.6 Inter-course Outcome Evaluations

Naturally, each course has its own model. However, in this section, we intend to measure inter-course performance evaluation where we train DOPE on one course and test it on another one. Table 2 shows the results. Again, the models are trained for the binary classification and they incorporate 20 weeks of the click data. Also, for the reference, we have included intra-course performance (i.e., the same course for training and test) shown in the diagonal entries of Table 2. Expectedly, when the training course and the test course are the same (i.e., intra-course setting), the performance is higher. This seems reasonable since clicking patterns are expected for the course in the past (i.e., a part of the training data) and the one in the future (i.e., testing data), and the model can more easily extract such patterns. Although the results for inter-course results are not as good as the ones for intra-course, we still see that the DOPE can effectively achieve reasonable performance. This indicates that the proposed model DOPE can detect salient click and demographic patterns that are transferable from a course to another.

Table 2: Inter-course performance evaluation

Training course	Test course					
	SS_1	SS_2	SS_3	ST_1	ST_2	ST_3
SS_1	**0.83**	0.78	0.77	0.71	0.8	0.75
SS_2	0.63	**0.80**	0.58	0.66	0.53	0.66
SS_3	0.64	0.51	**0.80**	0.45	0.72	0.49
ST_1	0.60	0.79	0.71	**0.82**	0.47	0.41
ST_2	0.74	0.60	0.56	0.62	**0.88**	0.49
ST_3	0.79	0.76	0.75	0.70	0.77	**0.86**

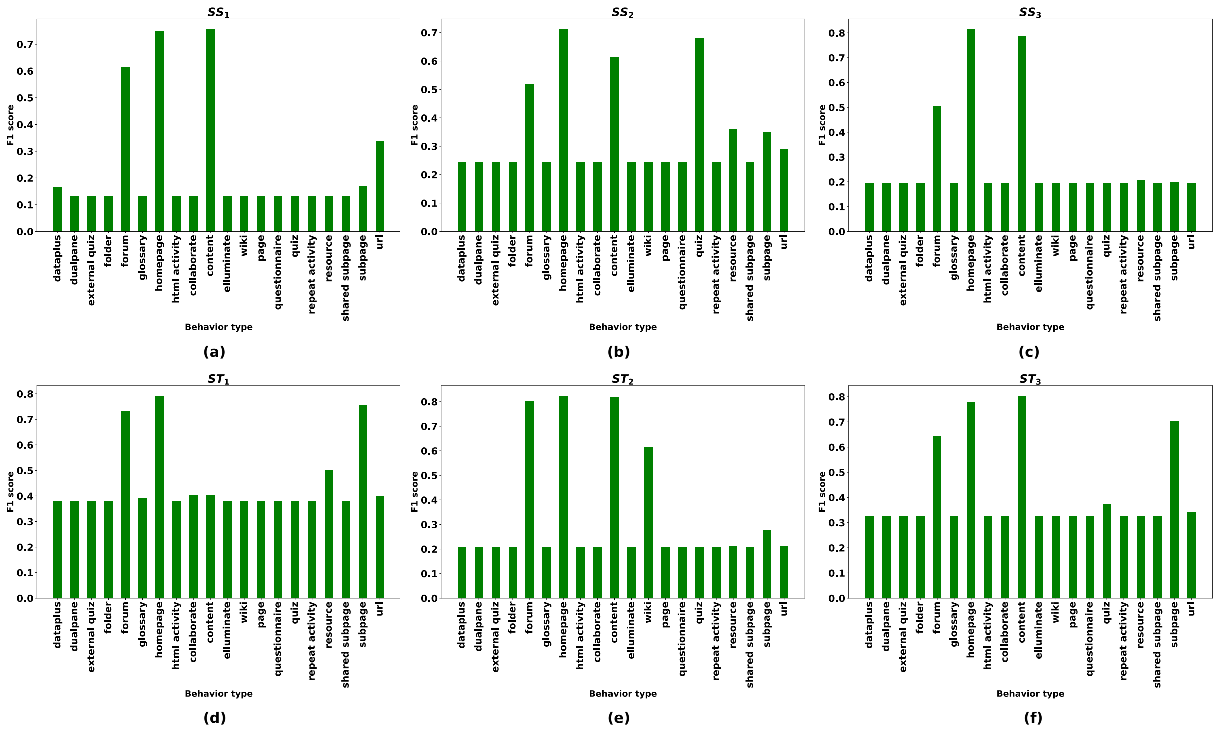


Figure 7: Behavioral feature analysis on different courses. The models are binary classification using 20 weeks of click data.

5. RELATED WORK

In the following, we highlight some of the works focusing on student dropout and performance prediction. In [35], they extracted 27 interpretive features and used logistic regression to predict student persistence prediction. The authors of [32] used probabilistic soft logic to model student survival by constructing probabilistic soft logic rules and associating them. Different from [32] (which mainly considered forum features), in [28] they did not consider forum data, but instead only made use of clickstream data to train their prediction model. More specifically, they used principal component analysis [37] paired with a linear support vector machine [4] for each week. It was in [12] that a more comprehensive approach was taken that used standard classification trees [2] and adaptive boosted trees [1] to construct their two-stage Friedman and Nemenyi procedure for dropout prediction by processing different features such as clickstream-based, forum-based, and assignment-based features. More recently, in [3], the authors studied a hybrid method for dropout prediction by combining both a decision tree [11] and extreme learning machine [18]. In addition to these traditional machine learning methods, some researchers have tried to use different deep learning models for dropout prediction of online courses. In [9] an LSTM was used to deal with the features extracted from students' interaction with lecture videos, forums, quizzes, and problems. [36] explored the potential benefits of employing a fully connected feed-forward neural network for dropout prediction. Different from previous work, [10] proposed a context-aware feature interaction network to incorporate context information of both participants and courses. More specifically, they used an attention-based mechanism for learning activity features. The most similar method to ours is found in [17] where they sought to conduct performance evaluations on students using a graph neural network (GNN), but there are primary differences: (1) they constructed separate small graphs of courses for each student while DOPE constructs a single knowledge graph of historical student course relations; (2) their graph neural network was used to obtain a graph classification for a given student based on that student's specific course graph, while our method uses the relational graph neural network to learn embeddings for both students and courses from a single large knowledge graph; and (3) DOPE furthermore utilizes an LSTM model to capture a student's rich sequential behavioral data beyond just using static fixed student features.

6. CONCLUSION AND FUTURE WORK

In this paper, we proposed a model for course performance prediction we call it Deep Online Performance Evaluation (DOPE). Our method first represents the online learning system as a knowledge graph, such that we then learn student and course embeddings from historical data using a relational graph neural network. Simultaneously, DOPE utilizes an LSTM for harnessing the student behavior data into a condensed encoding, as the data has a natural inherent sequential form. We tested the proposed model on six courses from the OULAD dataset where the results showed the feasibility of DOPE and that it can predict at-risk students of on-going courses. We also investigated the usefulness of the different types of behavioral features and observed that DOPE encodes the data in an intuitive manner.

In the future, we will first analyze the imbalance and sparse issues of the dataset. One possible way to alleviate the sparsity would be through a network alignment [6] of multiple MOOC datasets represented as knowledge graphs or connecting student behavior data from social media for better predictions in online education [22]. Also, we will investigate more advanced ways of handling behavioral data. For example, investigating better ways to use "subpage" clicks beyond a simple aggregation that ignores separating the multiple different "subpages". In addition, we plan to apply our framework to the traditional education system aiming at identifying similarities and differences between online and traditional course performance prediction, since we believe this to be highly important in improving online learning systems.

Acknowledgement

This research is supported by the National Science Foundation (NSF) under grant numbers IIS1907704, IIS1928278, IIS1714741, IIS1715940, IIS1845081, and CNS1815636.

7. REFERENCES

[1] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[2] W. Buntine. Learning classification trees. *Statistics and computing*, 2(2):63–73, 1992.

[3] J. Chen, J. Feng, X. Sun, N. Wu, Z. Yang, and S. Chen. Mooc dropout prediction using a hybrid algorithm based on decision tree and extreme learning machine. *Mathematical Problems in Engineering*, 2019, 2019.

[4] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[5] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[6] T. Derr, H. Karimi, X. Liu, J. Xu, and J. Tang. Deep adversarial network alignment. *arXiv preprint arXiv:1902.10307*, 2019.

[7] T. Derr, Y. Ma, W. Fan, X. Liu, C. Aggarwal, and J. Tang. Epidemic graph convolutional network. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, page 160–168. ACM, 2020.

[8] T. Derr, Y. Ma, and J. Tang. Signed graph convolutional networks. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 929–934. IEEE, 2018.

[9] M. Fei and D.-Y. Yeung. Temporal models for predicting student dropout in massive open online courses. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 256–263. IEEE, 2015.

[10] W. Feng, J. Tang, and T. X. Liu. Understanding dropouts in moocs. *Association for the Advancement of Artificial Intelligence*, 2019.

[11] J. Friedman, T. Hastie, R. Tibshirani, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000.

[12] J. Gardner and C. Brooks. Dropout model evaluation in moocs. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[13] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011.

[14] A. Graves, N. Jaitly, and A.-r. Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE workshop on automatic speech recognition and understanding*, pages 273–278. IEEE, 2013.

[15] C. He, P. Ma, L. Zhou, and J. Wu. Is participating in mooc forums important for students? a data-driven study from the perspective of the supernetwork. *Journal of Data and Information Science*, 3(2):62–77, 2018.

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

[17] Q. Hu and H. Rangwala. Academic performance estimation with attention-based graph convolutional networks. In *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, volume 168, pages 69–78. ERIC.

[18] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: a new learning scheme of feedforward neural networks. In *2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541)*, volume 2, pages 985–990. IEEE, 2004.

[19] N. Jha, I. Ghergulescu, and A.-N. Moldovan. Oulad mooc dropout and result prediction using ensemble, deep learning and regression techniques. 2019.

[20] K. Jordan. Massive open online course completion rates revisited: Assessment, length and attrition. *International Review of Research in Open and Distance Learning*, 16(3), 2015.

[21] H. Karimi*, T. Derr*, A. Brookhouse, and J. Tang. Multi-factor congressional vote prediction. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 266–273, 2019.

[22] H. Karimi, T. Derr, K. Torphy, K. Frank, and J. Tang. A roadmap for incorporating online social media in educational research. *Teachers College Record Year Book*, (2019), 2019.

[23] H. Karimi, P. Roy, S. Saba-Sadiya, and J. Tang. Multi-source multi-class fake news detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1546–1557, 2018.

[24] H. Karimi and J. Tang. Learning hierarchical discourse-level structure for fake news detection. In *Proceedings of the 2019 Conference of the NAACL-HLT, Volume 1*, pages 3432–3442, 2019.

[25] H. Karimi, J. Tang, and Y. Li. Toward end-to-end deception detection in videos. In *2018 IEEE International Conference on Big Data*, pages 1278–1283, 2018.

[26] H. Karimi, C. VanDam, L. Ye, and J. Tang. End-to-end compromised account detection. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 314–321, 2018.

[27] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[28] M. Kloft, F. Stiehler, Z. Zheng, and N. Pinkwart. Predicting mooc dropout over weeks using machine learning methods. In *Proceedings of the EMNLP 2014 workshop on analysis of large scale social interaction in MOOCs*, pages 60–65, 2014.

[29] J. Kuzilek, M. Hlosta, and Z. Zdrahal. Open university learning analytics dataset. *Scientific data*, 4:170171, 2017.

[30] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

[31] B. K. Pursel, L. Zhang, K. W. Jablokow, G. Choi, and D. Velegol. Understanding mooc students: motivations and behaviours indicative of mooc completion. *Journal of Computer Assisted Learning*, 32(3):202–217, 2016.

[32] A. Ramesh, D. Goldwasser, B. Huang, H. Daume III, and L. Getoor. Learning latent engagement patterns of students in online courses. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[33] J. L. Santos, J. Klerkx, E. Duval, D. Gago, and L. Rodríguez. Success, activity and drop-outs in moocs an exploratory study on the uned coma courses. In *Proceedings of the Fourth International Conference on Learning Analytics and Knowledge*, pages 98–102. ACM, 2014.

[34] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer, 2018.

[35] C. Taylor, K. Veeramachaneni, and U.-M. O'Reilly. Likely to stop? predicting stopout in massive open online courses. *arXiv preprint arXiv:1408.3382*, 2014.

[36] J. Whitehill, K. Mohan, D. Seaton, Y. Rosen, and D. Tingley. Delving deeper into mooc student dropout prediction. *arXiv preprint arXiv:1702.06404*, 2017.

[37] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[38] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*, 2019.

[39] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*, 2018.