Sequencing Educational Content Using Diversity Aware Bandits

Colton Botta
University of Edinburgh
cgb45@cam.ac.uk

Avi Segal
Ben-Gurion University
avise@post.bgu.ac.il

Kobi Gal
Ben-Gurion University
University of Edinburgh
kobig@bgu.ac.il

ABSTRACT
One important function of e-learning systems is to sequence learning material for students. E-learning systems use data, such as demographics, past performance, preferences, skillset, etc. to construct an accurate model of each student so that the sequencing of educational content can be personalized. Some of these student features are "shallow" traits which seldom change (e.g. age, race, gender) while others are "deep" traits that are more volatile (e.g. performance, goals, interests). In this work, we explore how reasoning about this diversity of student features can enhance the sequencing of educational content in an e-learning environment. By modeling the sequencing process as a Reinforcement Learning (RL) problem, we introduce Diversity Aware Bandit for Sequencing Educational Content (DABSEC), a novel contextual multi-armed bandit algorithm that leverages the dynamics within user features to cluster similar users together when making sequencing recommendations.

Keywords
Reinforcement Learning, Contextual Multi-Armed Bandit, Educational Sequencing

1. INTRODUCTION
Advancements in Artificial Intelligence (AI) have resulted in vastly improved models of student learning [4, 11, 14, 19]. Algorithms that use these models rely on data that describes students' online interactions, as well as their demographic information, previous academic performance, success on diagnostic questions, etc. All of this data can be collectively referred to as the *context* of the student, and it is within such contexts that algorithms operate in order to decipher how students are learning and how to best aid them. How these varying contextual features collectively model the complexities of human beings is of particular interest in this work, an idea we refer to as *human contextual diversity*. The advancement of e-learning technologies have brought together students of varied backgrounds and learning behaviors into single platforms, and reasoning about the diversity this creates when sequencing educational content is critical. We hypothesize that combining insights from social science about diversity can enrich educational models of students' behavior and improve the performance of educational sequencing algorithms. This work addresses the following questions: How can a machine detect human contextual diversity in educational data? Can we leverage the diverse and dynamic nature of this human data to improve how we sequence educational content to students?

To address these questions, we present a novel reinforcement learning algorithm, Diversity Aware Bandit for Sequencing Educational Content (DABSEC). DABSEC is a "diversity aware"[20] Contextual Multi-Armed Bandit (CMAB) algorithm with three main steps: calculate the dynamics of the underlying human contextual diversity in a group, form clusters of users with similar feature dynamics, and utilize these clusters and past student performance to sequence learning content to students. We compare the performance of DABSEC against LOCB [1], a state-of-the-art contextual bandit algorithm, as a baseline on two public educational datasets. Our results show that DABSEC achieves a higher average reward than LOCB on each dataset when predicting students' responses to questions.

2. BACKGROUND
We give an overview of CMAB algorithms and diversity.

2.1 Contextual Multi-Armed Bandits
Prior work has established that Bandit Algorithms, and RL in general, are effective solutions to educational sequencing[6]. One type of Bandit Algorithm, the Contextual Multi-Armed Bandit (CMAB) is a simplification of the full RL-problem and an extension of the Multi-Armed Bandit (MAB) problem where, at each timestep, the agent is presented with a list of arms (possible actions). Additionally, and unlike the original MAB setup, the agent is also presented with context (additional data) about the environment. The goal of the agent is to select a single arm, resulting in that action being performed. The agent then receives a reward for that arm only. Over time, the agent learns the underlying reward distribution of each arm and how that distribution is influenced by the context, and endeavors to maximize the total reward received over time [22].

CMABs have been used to sequence instructional material to students to increase overall learning [23, 15, 12], recom-

C. Botta, A. Segal, and K. Gal. Sequencing educational content using diversity aware bandits. In M. Feng, T. Käser, and P. Talukdar, editors, *Proceedings of the 16th International Conference on Educational Data Mining*, pages 502–508, Bengaluru, India, July 2023. International Educational Data Mining Society.

https://doi.org/10.5281/zenodo.8115731

mend news articles to readers [16], recommend the position of e-commerce items to maximize the chance a user interacts with them online [10], and many other use cases [3]. One recent work introduced the Local Clustering in Bandits (LOCB) algorithm [1] which implemented a "soft" clustering approach, by which users are clustered together if their preferences are within a certain threshold of each other. In this work, we use CMAB to select questions that students are most likely to get correct based upon their past question answering sequence.

2.2 Diversity

The existence of differences between humans in a group is one notion of diversity [2], with these differences often falling into two distinct categories: surface-level differences and deep-level differences [9]. Surface-level differences include, for example, age, sex, ethnicity, and race and are generally defined by their low-dynamics and ability to be observed immediately [13]. Deep-level differences, on the other hand, may include skills, values, preferences, and desires. These are more volatile and can only be observed through prolonged interaction between people [9]. For our purposes, we define *surface-level diversity* and *deep-level diversity* as differences between humans with respect to their surface-level and deep-level differences, respectively. One example of the importance of this classification is highlighted by the WeNet project, which places human diversity at the center of a new machine mediated paradigm of social interactions [2].

3. DABSEC

This section details the Diversity Aware Bandit for Sequencing Educational Content (DABSEC) algorithm.

3.1 Problem Definition

Assume $N = \{1, ..., n\}$ representing a set of n total users and $T = 1, ..., t$ representing a sequence of timesteps. At a timestep, t, a user, i_t, is drawn such that $i_t \in N$. Alongside i_t, the agent receives the context, $C_t = \{c_{1,t}, c_{2,t}, ..., c_{k,t}\}$ with one context vector for each of k arms and each context vector having dimension d such that $c_{k,t} \in \mathbb{R}^d$. The agent chooses one context vector, $c_{k,t}$, associated with arm $x_{k,t}$, to recommend to i_t and receives reward r_t in return. We assume that each user is associated with an unknown bandit parameter $\theta_{i,t}$ that describes how i_t interacts with the environment and can be thought of as a representation of how user i_t behaves [1]. As in previous bandit settings [16, 1, 7], the goal is to minimize the total regret, R_T given by:

$$R_T = \sum_{t=1}^{T} [\theta_{i,t}^{\intercal}(argmax_{c_{k,t} \in C_t} \theta_{i,t}^{\intercal} c_{k,t}) - \theta_{i,t}^{\intercal} c_t] \qquad (1)$$

where, at each round, t, we compute the regret by taking the reward achieved from the best possible arm choice, $x_{k,t}$, and subtracting the reward achieved from the agent's chosen arm, x_t. We also assume that each user, i, has a set of features, F, of length q such that at any time, t, there exists $F_{i,t} = \{f_{i,1,t}, f_{i,2,t} ..., f_{i,q,t}\}$.

3.2 DABSEC Algorithm

The DABSEC algorithm has three main steps: calculate the underlying feature dynamics of all users over time, form clusters of users with similar feature dynamics, then utilize the clusters and past student performance to sequence learning content to students. DABSEC (Algorithm 1) is initialized with the number of clusters to maintain (s), the frequency with which to update the clusters ($T_{cluster}$), the frequency with which to update the user feature dynamics ($\mathcal{U}$), and an exploration parameter (α). Then, all users are initialized (Lines 2-4) and the algorithm begins iterating over all timesteps sequentially (Line 5). In each round, t, a user i_t is presented along with the set of context vectors C_t (Line 6). DABSEC begins without any user clusters. DABSEC first checks if there are any clusters (Line 7), and if there are none (length($\mathcal{G} \leq 0$)), then the arm with the highest upper confidence bound (UCB) is chosen. As is standard practice [16] in bandit algorithms, UCB is computed using the estimation of user i_t's unknown bandit parameter, $\hat{\theta}_{i,t}$ (Lines 14-16) where $A_{i,t-1}^{-1}$ is the covariance matrix and $b_{i,t-1}$ is a normalizing matrix for user i at timestep $t-1$ that are used to compute the ridge regression solution of the coefficients [16]. On the other hand, if a user clustering has been established (length($\mathcal{G} > 0$)), then the cluster holding user i_t is set as $g_{s,t}$ (Line 8) and DABSEC calculates $\hat{\theta}_{g_{s,t}}$, which represents the unknown bandit parameter for the entire cluster (Line 9).

Finally, to choose an arm, we compare the UCB using the user's unknown bandit parameter, $\hat{\theta}_{i,t}$ to the UCB using the average unknown bandit parameter of all users in cluster $g_{s,t}$, $\hat{\theta}_{g_{s,t}}$ (Lines 10-12). The maximum of these two UCB values is selected (Line 13). The reasoning behind this is that previous work has established that clustering users by unknown bandit parameter is an effective strategy for identifying users who behave similarly in a task, thus resulting in a collaborative filtering effect [8, 7, 17, 18, 1]. In datasets where changes in user features are not available or considered, these past works still represent the state of the art in clustering bandit algorithms. Our approach, by comparison, is to gain an advantage in datasets where user feature dynamics are available and changing. In these cases, we expect the collective bandit parameter of the cluster where user i_t resides, $\hat{\theta}_{g_{s,t}}$, to estimate expected behavior better than $\hat{\theta}_{i,t}$.

With an arm chosen and pulled, we observe the reward, r_t, then update user parameters and cluster parameters for the cluster that user i_t resides in (Lines 17-22). Then, any user features, $F_{i,t}$ are updated (Lines 23-24). This step will be tailored to the specific implementation and dataset, as the number, type, and sophistication of the user features will be entirely dependent on the problem definition and setup. The count for how many times user i_t has been considered is also updated (Line 25). Finally, the most up to date clusters, $\mathcal{G}_t$, are calculated and returned by the CLUSTER function (Line 26 - see Algorithm 2), which ends round t.

3.3 Clustering by User Feature Dynamics

The second component of DABSEC is clustering users based upon the similarity of their feature dynamics. The CLUSTER algorithm (Algorithm 2) assumes that each user has a set of features, F, of length q such that at any time, t, there exists $F_{i,t} = \{f_{i,1,t}, f_{i,2,t} ..., f_{i,q,t}\}$. The values of each individual user feature, $f_{i,q,t}$ may change over time, which can be tracked to cluster users based upon the similarity of their feature dynamics. To do this, one can observe the value of a feature at some initial timestep, then again at a later timestep, and calculate the absolute value of the difference between them. More formally, at some initial timestep, $T_{initial}$, we store the values of all features for a given user, i_t: $F_{i_t,T_{initial}}$. We also initialize a set Y_t that contains one value for each user such that $Y_t = \{y_{1,t}, y_{2,t} ... y_{i,t}\}$ and $y_{i,t}$ represents the number of times that the agent has made a recommendation to user i_t. Thus, each time user i_t is selected by the algorithm, we can update $F_{i,t}$ based upon the observed user features at timestep t, and increment $y_{i,t}$ by 1. Once the agent has made a recommendation to a user $\mathcal{U}$ times, say at time T_{final}, such that $y_{i,t} = \mathcal{U}$, the feature dynamics for user i, δ_i, can be computed based upon how the features have changed between $T_{initial}$ and T_{final} (Algorithm 2 Line 2). The differences are summed over time to compute δ_i, and $\mathcal{U}$ is a hyperparameter that controls how often user feature dynamics are updated. After this calculation, $T_{initial}$ is set to T_{final} and $y_{i_t,t}$ is set to 0. The process repeats when $y_{i_t,t} = \mathcal{U}$ until all timesteps are complete.

By performing this operation for every user, we constantly have access to δ_i which represents the current dynamics of user i's features. We use the similarity between user's δ values to cluster them together, rather than $\theta_{i,t}$ as done in previous works [8, 7, 17, 18, 1]. To that end, we assume that there exists a set of clusters $\mathcal{G}$ of length s such that $\mathcal{G}_t = \{g_{1,t}, g_{2,t} ... g_{s,t}\}$. For simplicity, we assume that each user must appear in exactly one cluster and all users are split evenly amongst the clusters. This results in each cluster containing $\frac{n}{s}$ users. See Algorithm 2 for the full clustering pseudocode.

DABSEC updates clusters after a period of timesteps have passed $T_{cluster}$. This is because calculating the dynamics of the user features requires observing changes in those features over a period of time. To re-cluster after every timestep would not allow sufficient time to observe any true dynamics, so we update δ_i for each user after every $\mathcal{U}$ timesteps in which that user is selected.

Algorithm 1 DABSEC

Require: number of clusters to form s, cluster update frequency $T_{cluster}$, user feature dynamics update frequency $\mathcal{U}$, exploration parameter α

$T_{initial} \leftarrow 0$
for each $i \in N$ **do**
  $A_{i,0} \leftarrow I, b_{i,0} \leftarrow 0$
  $y_i \leftarrow 0$
for $t \leftarrow 1, 2 ... T_{final}$ **do**
  receive $i_t \in N$ and obtain $C_t = \{c_{1,t}, c_{2,t} ..., c_{k,t}\}$
  if length of $\mathcal{G} \geq 0$ **then**
    $g_{s,t} \leftarrow$ Cluster where i_t resides at round t
    $\hat{\theta}_{g_{s,t}} \leftarrow \frac{1}{|g_{s,t-1}|} \sum_{j \in g_{s,t-1}} A_{j,t-1}^{-1} b_{j,t-1}$
    $x_{cluster} \leftarrow argmax_{c_{a,t} \in C_t} \hat{\theta}_{g_{s,t}}^{\intercal} c_{a,t} + CB_{r,g_{s,t}}$ where $CB_{r,g_{s,t}} \leftarrow \frac{1}{|g_{s,t-1}|} \sum_{j \in g_{s,t-1}} \alpha \sqrt{c_{a,t}^{\intercal} A_{j,t-1}^{-1} c_{a,t}}$
    $\hat{\theta}_{i_t t} \leftarrow A_{i,t-1}^{-1} b_{i,t-1}$
    $x_{user} \leftarrow argmax_{c_{a,t} \in C_t} \hat{\theta}_{i_t t}^{\intercal} c_{a,t} + CB_{r,i}$ where $CB_{r,i} \leftarrow \alpha \sqrt{c_{a,t}^{\intercal} A_{i,t-1}^{-1} c_{a,t}}$
    $x_t \leftarrow max(x_{cluster}, x_{user})$
  else
    $\hat{\theta}_{i_t t} \leftarrow A_{i,t-1}^{-1} b_{i,t-1}$
    $x_t \leftarrow argmax_{c_{a,t} \in C_t} \hat{\theta}_{i_t t}^{\intercal} c_{a,t} + CB_{r,i}$ where $CB_{r,i} \leftarrow \alpha \sqrt{c_{a,t}^{\intercal} A_{i,t-1}^{-1} c_{a,t}}$
  pull x_t and observe reward r_t
  $A_{i,t} \leftarrow A_{i,t-1} + x_t x_t^{-1}$
  $b_{i,t} \leftarrow b_{i,t-1} + r_t x_t$
  if length of $\mathcal{G} \geq 0$ **then**
    $A_{g_{s,t},t} \leftarrow A_{g_{s,t},t-1} + x_t x_t^{-1}$
    $b_{g_{s,t},t} \leftarrow b_{g_{s,t},t-1} + r_t x_t$
  for $f_{i,q,t} \in F_{i,t}$ **do**
    update $f_{i,q,t}$ according to information gathered from problem setup and r_t
  $y_{i,t} \leftarrow y_{i,t} + 1$
  $\mathcal{G}_t \leftarrow CLUSTER(\mathcal{U}, Y, T_{cluster}, i_t)$

4. DABSEC ON EDUCATION DATA

In this section, we apply the DABSEC algorithm to two large-scale educational datasets: Eedi [24] and EdNet [5].

4.1 Eedi Dataset

Eedi[1] released a dataset that includes over 17 million interactions of students answering multiple choice questions. It was used for The NeurIPS 2020 Education Challenge [24] and contains two identically structured halves: Eedi1 and Eedi2. Each provides interaction logs of the student ID, question ID, student answer (range a-d), and the correct answer (range a-d). Every question has an associated list of features including a question ID, and a list of subject IDs (a list of IDs that correspond to mathematics concepts that are covered by the question). Every student has an associated list of features including gender, date of birth and a boolean indicator if the student is financially disadvantaged or not.

4.2 EdNet Dataset

The EdNet dataset[5] was the largest publicly-available education dataset when it was released in 2020. It contains over 131 million interactions from over 784,000 students who, over the course of two years, used the Santa[2] platform to study English for the Test of English for International Communication (TOEIC) exam. The dataset is organized in a 4-level, hierarchical style, and we consider the KT1 version for our analysis. The KT1 dataset is a collection of 784,309 CSV files, where each file contains the question answering logs of one student. Each line represents a question that the student answered, and includes the timestamp of the answer submission, a solving ID, the ID of the answered question, the student's answer (from a-d), and the amount of time spent answering the question. For each of the 13,169 questions in the dataset, the correct solution and the question tags are provided. These question tags are identical to the concept of subjects from the Eedi dataset described in section 4.1. We refer to the tags as subjects for consistency.

4.3 Experiments

In this section we describe an educational setting where an agent trained using DABSEC chooses personalized sequences of mathematics questions, based upon past student performance, that are likely to be answered correctly by the

[1] https://eedi.com

[2] https://www.aitutorsanta.com

Algorithm 2 *CLUSTER*

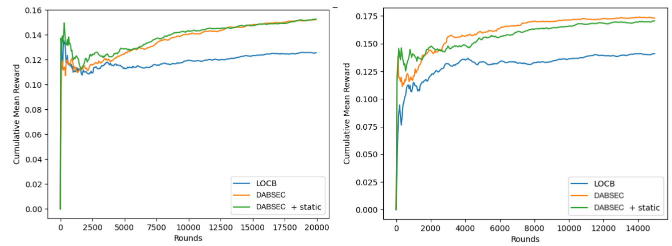
Require: user feature dynamics update frequency $\mathcal{U}$, user update counts Y, cluster update frequency $T_{cluster}$, user i_t

if $y_i == \mathcal{U}$ **then**
  $\delta_i = \sum_{q=1}^{Q}\{|F_{i,t} - F_{i,T_{initial}}|\}$
  $T_{initial} \leftarrow t$
  $y_i \leftarrow 0$
if t % $T_{cluster} == 0$ **then**
  $\delta_{sorted} \leftarrow$ sort δ in ascending order
  $\mathcal{G}_t \leftarrow$ split(δ_{sorted},s) where split(x,y) splits x into $length(x)\%y$ groups each of size $\frac{length(x)}{y} + 1$ and the rest of size $\frac{length(x)}{y}$
return $\mathcal{G}_t$

student. We apply DABSEC to Eedi1, Eedi2 and EdNet, by first obtaining the full list of unique questions that each student answered, along with the subject categories, the student answer, and correct answer for each question. At each round where user i_t is selected, we randomly sample 10 questions that student i_t has answered. Because we are interested in building an agent that can identify questions that each student should be able to answer correctly, we follow a recent approach [1] of selecting 9 questions that the student answered incorrectly in the past, and 1 question that the student answered correctly in the past. The correct question is not revealed to the agent. Not all students in the dataset answered enough total questions to be considered in this experimental setup, so we selected a subset: for the Eedi datasets, we consider the 50 users with the most total questions answered. For the EdNet dataset, we sample 50 users who have answered over 1000 questions. Thus, during each round of DABSEC, the agent receives a user, i_t, a list of 10 random questions that i_t has answered in the past (9 incorrect, 1 correct) and a context vector that contains the student's past performance by subject. The agent then chooses 1 question that it believes i_t is mostly likely to answer correctly. The agent is given a reward of 1 if it correctly selects the 1 question that user i_t did answer correctly in the past, and a reward of 0 otherwise. To compare the performance across datasets and against the baseline, we calculate and report the cumulative average reward achieved over every sequence of 50 timesteps.

Using the above setup, we first applied the original LOCB algorithm to both datasets. The creators released an open-source implementation of LOCB[3] which we extended and adapted to operate on our datasets. After the base setup, the algorithm continually forms and updates clusters based on the similarity of student's unknown bandit parameter, θ, which is a proxy for student preferences and behavior as discussed in Section 3. At each timestep, LOCB computes the average θ of the current student's cluster and uses it to select the question that was most likely answered correctly. In the original work's main experiments, the authors conclude that setting the number of clusters to 20, gamma to 0.2 and delta to 0.1 would return good results on average, so we use these values for our LOCB implementation.

[3]https://github.com/banyikun/LOCB



(a) Eedi1 Dataset (b) Eedi2 Dataset

Figure 1: A comparison of the performance of DABSEC, DABSEC + static, and LOCB on both Eedi datasets based on cumulative average reward.

We then applied DABSEC to all datasets, with clusters being continually updated every $T_{cluster}$ timesteps based on the average bandit parameter, θ, of a user's cluster, where clusters are formed based on similarity of feature dynamics as discussed in Section 3. We set the following hyperparameters for both datasets: $T_{cluster} = 1000$, $\mathcal{U} = 10$, and $s = 3$, as these produced the best overall performance. Additional hyperparameter settings are described in Appendix A.

Finally, for the Eedi dataset only, we follow an identical setup as DABSEC described above with the addition of the static (low dynamic) student features: the age, gender, and if they are financially disadvantaged. We call this DABSEC + static. We do not apply DABSEC + static to the EdNet dataset because there are no demographic features.

5. RESULTS AND ANALYSIS

We compare the performance of DABSEC, DABSEC + static, and LOCB on all datasets, and describe DABSEC's potential educational applications.

5.1 Results

As shown in Figure 1a, both of the DABSEC variations outperform the LOCB baseline by nearly 30% with respect to cumulative mean reward obtained over time on the Eedi1 dataset. Neither DABSEC variation seems to outperform the other. Looking at Figure 1b, we see that both DABSEC variations again outperform the LOCB baseline on the Eedi2 dataset - this time by about 25%. In this dataset, DABSEC slightly outperforms DABSEC + static but the gap is nearly closed by the time we reach the end of the rounds. Finally, in Figure 2, DABSEC outperforms the LOCB baseline by over 30% on the EdNet dataset.

Our experimental results confirm that DABSEC achieves better performance than LOCB on the Eedi1, Eedi2, and EdNet datasets. We found evidence that identifying and extracting feature dynamics can improve RL algorithm performance, and that clustering users based on their feature dynamics, rather than estimated user preferences alone, is a good starting towards improving clustering algorithms based on human diversity. We argue that the reason for this improvement is that identifying the highly dynamic features allows DABSEC to search the space of context-reward associations more completely and more quickly, thus leading to better reward. The low dynamic, static features, on the other hand, either exclude part of the search space or explore

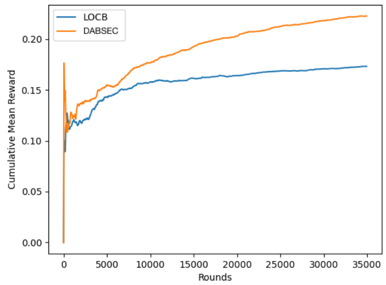


Figure 2: A comparison of the performance of DABSEC and LOCB on the EdNet dataset based on cumulative average reward. We ran 35,000 rounds until seeing evidence of stabilization.

it more slowly than DABSEC is capable of learning, leading to lower reward over the same timespan. This theory requires further testing, but the results of applying DABSEC to real data are promising, and further research into augmenting our clustering approach is planned for the future.

5.2 Implications for Education

We believe that a diversity aware approach to RL has high potential in the education domain. Due to the amount of individual behavioral data, one of the dominant use cases of RL and Bandit algorithms is e-learning systems, where students answer questions while the system attempts to observe, understand, and improve student knowledge based on the responses [21]. This is an ideal environment where user features are highly dynamic, as student performance across subjects changes with each question answered. This is a phenomenon we saw in our experiments in Section 4 and were able to exploit to boost performance. We believe that there is a potential for algorithms like DABSEC to further improve e-learning technology.

6. CONCLUSION

In this work, we designed, implemented, and tested DABSEC, a diversity aware RL algorithm that uses feature dynamics as a proxy for underlying human-contextual diversity, then clusters users based on this metric. We hypothesized that this technique could improve RL algorithms that operate in environments where user data is highly dynamic, and this proved true when applying DABSEC to two large-scale educational datasets. DABSEC outperforms the LOCB baseline by approximately 30% based on cumulative mean reward earned over time, and we believe that extensions to DABSEC can make it an ideal tool for building more performant e-learning applications.

6.1 Limitations

Our approach is an initial attempt to develop a diversity aware RL approach that leverages the dynamics of human data over time. One major drawback is that if a dataset is mostly comprised of features with low dynamics, the user feature dynamics would always be calculated as near zero and the clusters would be far less informative. Similarly, our assumption that user's could only be in one cluster may fall short of fully capturing the most available data on every student, as LOCB found by letting user's reside in multiple clusters simultaneously [1]. Similarly, by requiring all clusters to include the same number of users, we may not be forming the ideal clusters - for example, if the cluster size dictates that each cluster should have 10 users, but there are 3 users that are extreme outliers, then these 3 might benefit from residing in their own cluster. Additionally, in our definition of diversity, we assume that user features that remain constant are likely surface-level, whereas more dynamic features are likely deep-level. Of course, this may not hold in all situations; some people's goals, personalities, and values may never change, despite being classified as traits of deep-level diversity. For the sake of this work, we make this assumption based upon past sociology research [9, 13], but acknowledge that it may not hold in all implementation use cases. Finally, we followed the experimental approach that LOCB[1] used by randomly selecting the data at each round - we picked the student randomly, then randomly chose 9 questions that the student got incorrect and 1 that the student got correct to serve as the arms. This assumes knowledge of the entire dataset at the beginning, which would not be the case in real-time e-learning systems which consider student interactions as they occur.

6.2 Future Work

Further research should be conducted to improve upon our initial findings. First, there is an opportunity to improve the clustering algorithm to account for additional data about the user. For example, users could be clustered using a combination of overall feature dynamics and the preferences of users, represented by their unknown bandit parameter θ. This technique may boost performance by clustering users based upon both their preferences and how those preferences are changing over time. Second, this work included running DABSEC on two real-world educational datasets, but deploying DABSEC in the wild would offer further insight into the usefulness of diversity-aware RL. We would like to deploy DABSEC in a live e-learning platform so that it can sequence learning content to students in real-time. Finally, given that incorporating human data and diversity within algorithms needs to be handled with care, an exciting extension of this work would be to consider if diversity-aware algorithms have any implications on algorithmic fairness. For instance, investigating whether or not algorithmic fairness is more easily achieved with a diversity-aware algorithm, or if diversity-aware algorithms are more or less transparent than traditional algorithms are both important research areas to explore.

7. ACKNOWLEDGEMENTS

This work was supported in part by the European Union Horizon 2020 WeNet research and innovation program under grant agreement No 823783.

8. REFERENCES

[1] Y. Ban and J. He. Local clustering in contextual multi-armed bandits. In *Proceedings of the Web Conference 2021*, pages 2335–2346, 2021.

[2] I. Bison, M. Bidoglia, M. Busso, R. C. Abente, M. Cvajner, M. D. R. Britez, G. Gaskell, G. Sciortino, S. Stares, et al. D1. 3 final model of diversity: Findings from the pre-pilots study. 2021.

[3] D. Bouneffouf, I. Rish, and C. Aggarwal. Survey on applications of multi-armed and contextual bandits. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

[4] B. Choffin, F. Popineau, Y. Bourda, and J.-J. Vie. Das3h: modeling student learning and forgetting for optimally scheduling distributed practice of skills. *arXiv preprint arXiv:1905.06873*, 2019.

[5] Y. Choi, Y. Lee, D. Shin, J. Cho, S. Park, S. Lee, J. Baek, C. Bae, B. Kim, and J. Heo. Ednet: A large-scale hierarchical dataset in education. In *International Conference on Artificial Intelligence in Education*, pages 69–73. Springer, 2020.

[6] S. Doroudi, V. Aleven, and E. Brunskill. Where's the reward? a review of reinforcement learning for instructional sequencing. *International Journal of Artificial Intelligence in Education*, 29:568–620, 2019.

[7] C. Gentile, S. Li, P. Kar, A. Karatzoglou, G. Zappella, and E. Etrue. On context-dependent clustering of bandits. In *International Conference on Machine Learning*, pages 1253–1262. PMLR, 2017.

[8] C. Gentile, S. Li, and G. Zappella. Online clustering of bandits. In *International Conference on Machine Learning*, pages 757–765. PMLR, 2014.

[9] D. A. Harrison, K. H. Price, and M. P. Bell. Beyond relational demography: Time and the effects of surface-and deep-level diversity on work group cohesion. *Academy of management journal*, 41(1):96–107, 1998.

[10] X. He, B. An, Y. Li, H. Chen, Q. Guo, X. Li, and Z. Wang. Contextual user browsing bandits for large-scale online mobile recommendation. In *Fourteenth ACM Conference on Recommender Systems*, pages 63–72, 2020.

[11] J. He-Yueya and A. Singla. Quizzing policy using reinforcement learning for inferring the student knowledge state. *International Educational Data Mining Society*, 2021.

[12] W. Intayoad, C. Kamyod, and P. Temdee. Reinforcement learning based on contextual bandits for personalized online learning recommendation systems. *Wireless Personal Communications*, 115(4):2917–2932, 2020.

[13] S. E. Jackson, V. K. Stone, and E. B. Alvarez. Socialization amidst diversity-the impact of demographics on work team oldtimers and newcomers. *Research in organizational behavior*, 15:45–109, 1992.

[14] K. R. Koedinger, J. C. Stamper, E. A. McLaughlin, and T. Nixon. Using data-driven discovery of better student models to improve student learning. In *International conference on artificial intelligence in education*, pages 421–430. Springer, 2013.

[15] A. S. Lan and R. G. Baraniuk. A contextual bandits framework for personalized learning action selection. In *EDM*, pages 424–429, 2016.

[16] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670, 2010.

[17] S. Li, W. Chen, and K.-S. Leung. Improved algorithm on online clustering of bandits. *arXiv preprint arXiv:1902.09162*, 2019.

[18] S. Li, A. Karatzoglou, and C. Gentile. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 539–548, 2016.

[19] H. Nakagawa, Y. Iwasawa, and Y. Matsuo. Graph-based knowledge tracing: modeling student proficiency using graph neural network. In *2019 IEEE/WIC/ACM International Conference On Web Intelligence (WI)*, pages 156–163. IEEE, 2019.

[20] L. Schelenz, I. Bison, M. Busso, A. De Götzen, D. Gatica-Perez, F. Giunchiglia, L. Meegahapola, and S. Ruiz-Correa. The theory, practice, and ethical challenges of designing a diversity-aware platform for social relations. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 905–915, 2021.

[21] A. Singla, A. N. Rafferty, G. Radanovic, and N. T. Heffernan. Reinforcement learning for education: Opportunities and challenges. *arXiv preprint arXiv:2107.08828*, 2021.

[22] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[23] C. Tekin, J. Braun, and M. van der Schaar. etutor: Online learning for personalized education. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5545–5549. IEEE, 2015.

[24] Z. Wang, A. Lamb, E. Saveliev, P. Cameron, Y. Zaykov, J. M. Hernández-Lobato, R. E. Turner, R. G. Baraniuk, C. Barton, S. P. Jones, et al. Instructions and guide for diagnostic questions: The neurips 2020 education challenge. *arXiv preprint arXiv:2007.12061*, 2020.

APPENDIX

A. HYPERPARAMETER VARIATIONS

Using the DABSEC algorithm on the EdNet dataset, we also explored a few variations of the hyperparameters: the number of clusters to sort users into, s, the user feature dynamics update frequency, $\mathcal{U}$, and the cluster update frequency, $T_{cluster}$. Like before, we measure the performance based on cumulative mean reward achieved over time. Figure 3a shows the effect of changing the frequency with which the user feature dynamics are updated ($\mathcal{U}$). We held the number of clusters constant at 3 and the cluster update frequency constant at 1000. We set $\mathcal{U}$ as 5, 10, 50, and 100, which represent how many questions need to be answered by a user before we recalculate their current feature dynamics. We can see that the performance of DABSEC is not effected much by changing $\mathcal{U}$, though the best performing variation updated a user's feature dynamics after every 100 questions answered by that user. This makes sense, because a larger $\mathcal{U}$

forces a larger amount of questions to be answered between feature dynamics calculations, meaning that there will be far more data to consider than when $\mathcal{U}$ is smaller. However, the difference in performance is not very significant.

Figure 3b shows the effect of changing the frequency with which the actual global clusters are updated ($T_{cluster}$). We held the number of clusters constant at 3 and the user feature dynamics update frequency constant at 10. We set $T_{cluster}$ as 500, 1000, 2000, and 5000, which represent how many rounds occur between every instance of reclustering. We can see that the performance of DABSEC is not effected much by changing $T_{cluster}$, though the worst performing variation updated clusters every 500 rounds. This makes sense, because a smaller $T_{cluster}$ would not be considering as much data when forming new clusters, which may result in clusters that are less indicative of true similarities between users. It would make sense that a higher $T_{cluster}$ would result in more data being considered by the clustering algorithm, thus resulting in better clusters and a better performing algorithm. However, the difference in performance is not very significant.

Finally, Figure 4 shows the effects of changing the number of clusters that users are placed into. DABSEC achieves better performance when the number of clusters is smaller (3), with performance incrementally worsening as the number of clusters increases to 5, 10, and 15. This is in line with our expectations, as we are only using 50 total users which makes the size of the clusters quite small as the number of clusters increases. In the future, running these experiments with more total users would be interesting.

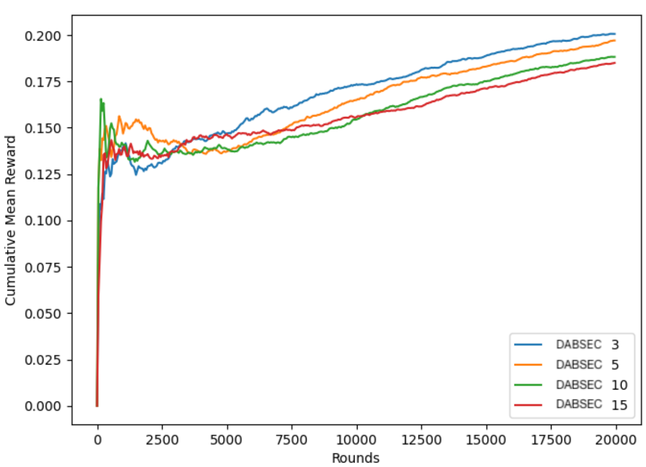
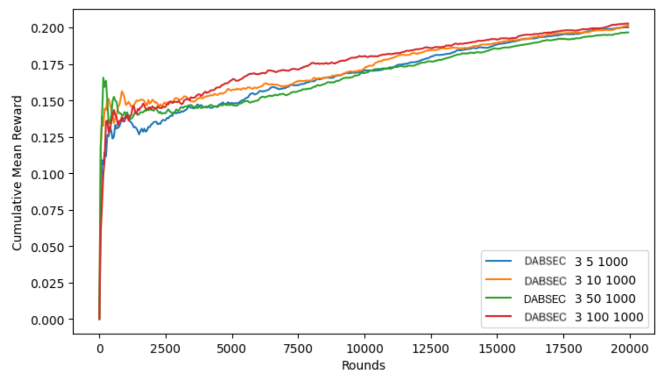
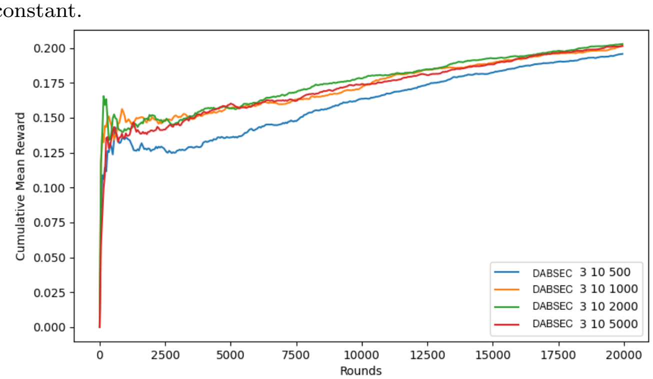


Figure 4: Cluster size varies (3, 5, 10, 15) while $T_{cluster}$ (1000) and feature dynamics update frequency (10 rounds) remains constant.



(a) Frequency of calculating user feature dynamics, δ, varies (5, 10, 50, 100 rounds) while clusters (3) and $T_{cluster}$ (1000 rounds) remain constant.



(b) Frequency of calculating the global clusters, $T_{cluster}$, varies (500, 1000, 2000, 5000 rounds) while clusters (3) and feature dynamics update frequency (10 rounds) remain constant.

Figure 3: Hyperparameter variations using DABSEC on the EdNet dataset.