DASH - Dynamic Adaptive Study Path Recommendation System Based on KT and ODT

SONG Shiyuan, WANG Hesong, ZHANG Chenyu
The Hong Kong University of Science and Technology

Abstract

This paper presents DASH, a study path recommendation system based on online decision transformers (ODT). DASH addresses the challenge of unstructured data from user-uploaded teaching videos by employing text embedding and similarity measures for topic clustering and performing the study path recommendation with reinforcement learning. The system's reinforcement learning framework utilizes learning cells as actions and a decision transformer equipped with online fine-tuning. A reward function based on the LPKT knowledge tracing model tracks learner progress, while cognitive navigation techniques and a knowledge graph enhance recommendation results. The LPKT model and reward functions are validated, with offline pretraining of the Decision Transformer conducted using various datasets. Future work includes exploring advanced clustering algorithms and employing interpretable knowledge tracing models. DASH aims to deliver personalized and optimized learning experiences through the integration of reinforcement learning techniques and decision transformers.

1. Introduction

Online learning platforms have revolutionized education by offering a wide range of learning resources, combining various media formats: texts, videos, and images. These platforms, such as Khan Academy [4], provide learners with adaptable and personalized learning experiences. Adaptive learning techniques are employed to tailor the learning materials and paths to individual learners, taking into account their current knowledge level and the relationships between different learning items. This paper aims to explore the effectiveness and benefits of online learning platforms with diverse media learning resources, highlighting their impact on enhancing the learning experience and outcomes for students.

The learning platforms have combined media data with text, video and images in order to achieve a better learning efficiency for audience. Deploying such a model on open learning platform has several challenges, the overall use of local deployment analysis of multi-modal models to extract corpus information from multiple media is expensive, because user upload operations are always performed at high frequency, so we use image caption techniques to mine the vector representation of various media information. We found that some models that reference both knowledge content and knowledge graphs has shown reliable recommendation result (Liu et al. [6]). However, compared with fixed course websites, it is difficult to obtain the relationship between knowledge graphs and media content on an open platform. Therefore, we use DNN to generate the topic relationship graph where the topic being clustered by k-mean algorithm. Optimize the relationship representation of the graph in the recommendation process and user feedback, and dynamically build the knowledge graph. We deploy the Decision Transformer [2] to offline pre-trained possible learning path recommendations, and then use the Online Decision Transformer [8] to perform online fine-tuning for specific students. Compared with traditional online reinforcement learning methods, this technique can reduce the risk of unexpected path recommendations. Smaller and reduces the risk of students as actors encountering wrong learning path guidance during the model update process and alleviating the sparse reward problem.

2. Related Work

Currently, there exists learning recommendation using collaborative filtering (e.g., KNN, MPR) and deep learning (e.g., GRU4Rec [5]). These methods aim to solve the sequence recommendation problem.

To further enhance the recommendation strategy, the knowledge tracing model (KT) is used to simulate student expected learning states to provide more accurate or interpretive path predictions. One approach from Chen et al. [3] and Tang et al. [7] is to model the evolution of knowledge level using techniques like the transition matrix in Markov Decision Process (MDP).

Another approach utilizes the knowledge structure itself to make recommendations (Zhu et al. [9]), such as applying path generation rules based on expertise or using semantic inference on ontology. Cai et al. [1] utilizes A2C reinforcement learning model to dynamically reallocate the learning cost when performing the learning path recommendation.

However, the design of reward function does not consider the distance between user's mastery status and the learning target. It is only focusing on the growth of the comprehensive overall knowledge level, without paying attention to whether the user has reached the expected level of ability mastery in a specific field, which leads to the non-personalized recommendation result when recommending paths to learners with clear learning goals. Most methods either consider the importance of knowledge level or the knowledge structure alone, without effectively combining both aspects. CSEAL (Liu et al. [6]) have established the cognitive structure rely on the designated knowledge graph. However, when facing the large variety of multimedia resources randomly uploaded by users, it is difficult for experts to construct a reliable knowledge graph to connect vast online resources.

3. Algorithm Outline

3.1. Topic Clustering

To organize and analyze diverse media assets on the open learning platform, a multi-modal approach is employed. Text-based assets, such as articles, are transformed into word embeddings using a pre-trained model. First, video and image assets are processed using an image captioning model, generating textual descriptions. Then by concatenating caption embeddings and reducing dimensionality with one layer of DNN for similarity comparison alignment. Similarity measures are then applied to quantify similarities between the encoded text-based and image caption embeddings.

K-means clustering groups media assets into topics based on similarity. This creates distinct clusters representing different themes. The topic organization forms a structured framework for understanding and categorizing media assets. A knowledge graph is generated for the topics using the KNN algorithm and word embeddings. This graph captures interconnections and dependencies, facilitating a holistic understanding. The study path recommendation system uses the graph to provide personalized learning paths. By considering topic relationships and word embedding similarity, the KNN algorithm identifies related topics, ensuring a tailored learning experience based on individual interests and goals.

The following Figure 1 shows that the procedure of topic clustering and the rational graph generation:

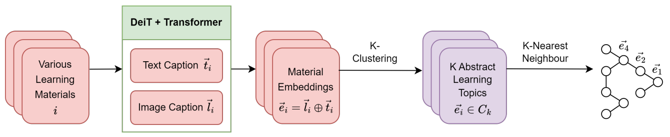
Various Learning Materials i
DeiT + Transformer
Text Caption $\vec{t}_i$
Image Caption $\vec{l}_i$
Material Embeddings $\vec{e}_i = \vec{l}_i \oplus \vec{t}_i$
K-Clustering
K Abstract Learning Topics $\vec{e}_i \in C_k$
K-Nearest Neighbour
$\vec{e}_4$
$\vec{e}_2$
$\vec{e}_1$

Figure 1. The procedure of topic clustering

3.2. Offline Reinforcement Learning

To represent students within the study path recommendation system, learning cells are defined as actions within the system and encompass various parameters, including the learning material embedding $e_i \in R^{d_e}, E = [e_1, e_2, \cdots, e_J] \in R^{J \times d_e}$, time spent on learning the material $a_t \in R^{d_k}$, interval time between learning materials $i_t \in R^{d_k}$, and grade $o_t \in \{\mathbf{1}, \mathbf{0}\}, o_t \in R^{d_o}$. Sequential data of users' learning history is collected to gain insights into their past learning experiences as $s_T = [e_{t-1}, a_{t-1}, o_{t-1}, i_{t-1}, e_t, a_t, \cdots, e_{T-1}, a_{T-1}, o_{T-1}]$.

In the offline reinforcement learning process of the decision transformer, we take the learning state $h_t = \text{LPKT}(s_T, h_{t-1})$ as the state s_t, and the choice of the next learning material e_t is regarded as the action. The reward of action at time t_k can be computed in a format of return to go $g_{t_k} = \sum_{t=1}^{t_k} r(e_t, s_t)$. In the following section, we will further derive $r_t = r(e_t, s_t)$, which is the reward of the action e_t under s_t at time t.

Offline reinforcement learning is employed to validate the algorithms before applying them in an online setting. This approach helps reduce the risk of exploration for students during the online algorithm iteration, ultimately minimizing instances where students may receive incorrect or sub-optimal learning paths.

3.2.1 Reward Function

As part of the offline reinforcement learning process, a knowledge tracing model called LPKT (Learning Process-consistent Knowledge Tracing) is trained for computing the reward for each selection of the next learning material. By leveraging the LPKT model, the reward function captures the learner's progress and informs the reinforcement learning process, ensuring that study path recommendations are aligned with the learner's goals and maximizing their learning outcomes.

In LKPT model, after completing the exercise e_t, the students' mastery level is updated by $h_t = \tilde{\text{LG}}_t + L_{ft}^G \cdot h_{t-1}$ where the $\tilde{\text{LG}}_t$ and L_{ft}^G are the learning gain and the forget gate respectively.

To formulate the learning gain and the forget gate, we first compute the answer embedding from user's learning history by partitioning the sequence to learning cells $c_t = [e_t, a_t, o_t]$. The answer embedding is one layer DNN $I_t = W_1^\top [e_t \oplus a_t \oplus o_t] + b_1$. For preparing to compute the learning gain controlled by learning gate, the raw learning result at time t is formulated by $I_{g_t} = \tanh(W_2^\top [I_{t-1} \oplus i_t \oplus I_t \oplus \tilde{h}_{t-1}] + b_2)$ where the $\tilde{h}_{t-1}$ is the result of student's mastery levels at time $t-1$, which is the result of the previous step. The learning gain then can be calculated by $\tilde{\mathrm{LG}}_t = q_{e_t} \cdot L_{gt}^G \cdot \left(\frac{1+I_{gt}}{2}\right)$ where the q_{e_t} restricts the learning gain matrix $\mathrm{LG}_t = L_{gt}^G \cdot \left(\frac{1+I_{gt}}{2}\right)$ to only update columns that related to topics of the learning material e_t, and the L_{gt}^G is the learning gate computed by $L_{gt}^G = \sigma(W_3^\top I_{gt} + b_3)$. On the other hand, the forget gate is formulated by $L_{ft}^G = \sigma(W_4^\top [h_{t-1} \oplus \tilde{\mathrm{LG}}_t \oplus i_t] + b_4)$ which takes the learning gain $\tilde{\mathrm{LG}}_t$ and the interval time i_t into consideration.

After formulating the two gates, the student's mastery levels on topics can be computed by $h_t = \tilde{\mathrm{LG}}_t + L_{tf}^G h_{t-1}$, and the mastery levels only related to the corresponding learning state to e_{t+1} can be obtained by $\tilde{h}_t = q_{e_{t+1}} \cdot h_t$.

The reward function is designed for considering both the growth of the H-matrix (learner's mastery of different k topics) and its distance from the learning goal. The following reward function is used in our offline decision transformer pre-training, where the h_t stands for the matrix of student's mastery levels on topics, $\tilde{h}_F = qe_{\mathrm{tgt}} h_{\max}$ represents the target mastery level, which is the student's mastery level matrix with highest possible value on the target learning topics, α is the parameter for balancing:

$$r'_t = \alpha(h_{t+1} - h_t) + (1-\alpha)(\tilde{h}_F - \tilde{h}_t)$$

Referring to the previous learning path recommendation reward function design in KT-KDM model (Cai et al. [1]), we also adopted the decay strategy for high reward recommendation results to avoid the model to repeatedly recommending the same learning materials with high rewards:

$$r_{k,t} = \begin{cases} \dfrac{r'_{k,t}}{n_{k,t}} & r'_{k,t} > 0 \\ r'_{k,t} & r'_{k,t} \le 0 \end{cases}$$

k represents the index of latest recommended learning material e_k, and $n_{k,t}$ represents the times the learning material e_k has been recommended in the learning path before time t.

By first performing offline reinforcement learning, the study path recommendation system can mitigate the risks associated with exploration and refine its algorithms before applying them in an online setting. This iterative process, with the LPKT model's involvement in the reward computation, helps ensure that students receive accurate and effective learning paths tailored to their needs and goals.

3.2.2 Algorithm Procedure

Here is the pseudo code for the offline reinforcement training and testing process, showing the workflow of our algorithm, see Figure 3 for more expressive information.

Algorithm 1 Offline RL Training and Testing

```
Require: Learning Material Embedding E, pre-trained LPKT model, train set TS, test set VS, Rational Graph K={$C, G, T$}
 1: Set train set with reward TSR = $\emptyset$
 2: for $s_T$ in Batch of TS do
 3:     Set H-matrix $h_t = h_0$ current step $t = 0$
 4:     while $t \le l$ do
 5:         Pick the learning cell $c_t$ from $s_T$
 6:         Calculate $h_{t+1} \leftarrow$ LPKT($c_t$, $h_t$)
 7:         Pick the action $e_t$ and index $k$ of $e_t$ from $c_t$
 8:         Calculate reward $r_{k,t}$ with $h_t$ and $h_{t+1}$
 9:         Calculate return to go $g_{t,k} \leftarrow \sum_{t=1}^{t_k} r_{k,t}$
10:         Add $[R_t, s_t, a_t] \leftarrow [g_t, h_t, e_t]$ to TSR
11:     end while
12:     back-propagation(transformer, Batch of TSR)
13: end for
14: for $s_T$ in VS do
15:     Input: Returns $R$, states $s$, actions $a$, timesteps $t$
16:     Output: Action predictions $a_{\text{preds}}$
17:     $R \leftarrow$ initialize to large value
18:     position embedding $\leftarrow \text{embed}_t(t)$
19:     input embeds $\leftarrow [\text{embed}(R), \text{embed}(s), \text{embed}(a)]$
20:     hidden states $\leftarrow$ transformer(input embeds)
21:     hidden $a \leftarrow$ unstack(hidden states).actions
22:     $\text{cog}_a \leftarrow \text{pred}_a(\text{hidden } a) \bigcap$ CN($C, G, T$)
23:     Update K with path $\{a_0, a_1, \cdots, a_{\text{pred}}\}$ under rate $p$
24:     return e: $\text{cog}_a$
25: end for
```

3.3. Cognitive Navigation (CN)

After obtaining the recommendation result from decision transformer, we refine the learning path predictions using the knowledge graph to further alleviate the unreasonable recommendation of the learning path. Apply cognitive navigation techniques inspired by the CSEAL (Liu et al. [6]) structure can help the model to leverage experiences from previous results that has been testified helpful on student's ability improvements since we only consider those result came from neighbours in the knowledge graph. The algorithm runs following the steps shown in Algorithm 1:

Algorithm 2 Cognitive Navigation

```
Require: Central focus $C$, prerequisite graph $G$, learning target $T$
Ensure: All $d \in D$ have a path to $T$ in $G$ and are one of the $k$-hop neighbors of $C$
 1: Initialize candidates $D = \emptyset$, queue $Q = \emptyset$
 2: Add $C$ to $D$
 3: Add successors within $k-1$ hops of $C$ to $D$
 4: Add predecessors within $k-1$ hops of $C$ to $Q$
 5: while $Q \neq \emptyset$ do
 6:     $q \leftarrow Q$.pop()
 7:     Add $q$ to $D$
 8:     Add neighbors of $q$ to $D$
 9: end while
10: for $d$ in $D$ do
11:     if $d$ cannot reach $T$ then
12:         Delete $d$ from $D$
13:     end if
14: end for
15: return $D$
```

4. Experiment

4.1. Data Processing

The KTM implemented in our experiment was trained on the ASSITment 2009-2010 dataset. This dataset was sourced from the online teaching system called ASSISTment, which provided teaching data for mathematics covering grades 4 to 10 during the period of 2009 to 2010. After removing any invalid data, we retained a total of 300,000 records in general for the training dataset. This dataset comprised of data from 4,417 students and covered 110 knowledge components (KCs).

To run the experiment, a single NVIDIA GeForce RTX 3090 GPU was utilized. The training process consisted of 40 epochs. During the pre-training on LPKT model, the AUC score reached a stopping point at 0.7952.

In the training process, the online decision transformer was not executed since it is specifically designed for online deployment scenarios. However, in the offline training process, the model is pre-trained using the offline decision transformer. This ensures that the model has the necessary knowledge and decision-making capabilities before it is deployed in an online setting.

4.2. Evaluation and Validation

In Table 1, the evaluation metrics KSS and KES are used to assess the models. The leftmost column lists the models, while the corresponding results are presented in the KSS and KES columns. We observe that the overall improvement in the H matrix is not significantly higher compared to the CSEAL model. This is due to our reward function considering both the increase in EP and the distance to the learning target. However, there is a potential bias in our training process as we set the learning target of students simply as their final result in the training set. The inclusion of knowledge graph generation and cognitive navigation techniques has improved the model compared to traditional approaches. Nevertheless, the automatically generated knowledge graph did not outperform the CSEAL model. It is important to note that the vast media platform makes it challenging to design a well-defined knowledge graph to link the learning materials for the CSEAL setup.

Table 1. Overall results of EP

Model	KSS	KES
KNN	0.000700	0.257919
GRU4Rec	0.007727	0.201219
MC-50	0.108636	-0.005103
DQN	0.100610	0.002688
CN-Random	0.272784	0.138526
CSEAL	0.346883	0.405823
DASH	0.220262	0.350243

In Figure 2, a comprehensive comparison between the two models was conducted. Although our target Ep did not reach a higher position, this can be attributed to factors such as the selection of different Knowledge Tracing Models and the emphasis of the reward function, which also affects the convergence speed to some extent. Our analysis showed that the covariance of our H matrix was larger, indicating a focus on learning goals rather than overall improvement in Ep (the promotion of knowledge level). These findings underscore the complexity of model evaluation and suggest that multiple factors contribute to the observed outcomes.

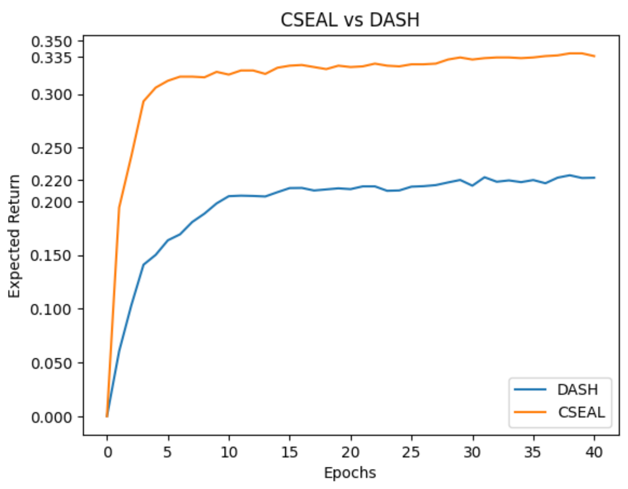

Figure 2. The Ep growth comparison between CSEAL and DASH

5. Future Work

Explore more powerful clustering algorithms other than k-mean for tagging multiple topics. Incorporate knowledge tracing models with higher interpretability, such as HKAT, for explainable recommendation results. Address the limitation of the Decision Transformer in handling multi-dimensional input generated by LPKT with more dimensions.

6. Appendix

References

[1] Dejun Cai, Yuan Zhang, and Bintao Dai. Learning path recommendation based on knowledge tracing model and reinforcement learning. *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*, pages 1881–1885, 2019. 2, 3

[2] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling, 2021. 1

[3] Penghe Chen, Yu Lu, Vincent Wenchen Zheng, and Yang Pian. Prerequisite-driven deep knowledge tracing. *2018 IEEE International Conference on Data Mining (ICDM)*, pages 39–48, 2018. 1

[4] Khan Academy: Free Online Courses, 2015. Accessed: 2023-11. 1

[5] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks, 2016. 1

[6] Qi Liu, Shiwei Tong, Chuanren Liu, Hongke Zhao, Enhong Chen, Haiping Ma, and Shijin Wang. Exploiting cognitive structure for adaptive learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery; Data Mining*, KDD '19. ACM, July 2019. 1, 2, 3

[7] Xueying Tang, Yunxiao Chen, Xiaoou Li, Jingchen Liu, and Zhiliang Ying. A reinforcement learning approach to personalized learning recommendation systems. *British Journal of Mathematical and Statistical Psychology*, 72:108–135, 2018. 1

[8] Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer, 2022. 1

[9] Haiping Zhu, Feng Tian, Ke Wu, Nazaraf Shah, Yan Chen, Yifu Ni, Xinhui Zhang, Kuo-Ming Chao, and Qinghua Zheng. A multi-constraint learning path recommendation algorithm based on knowledge map. *Knowl. Based Syst.*, 143:102–114, 2017. 1

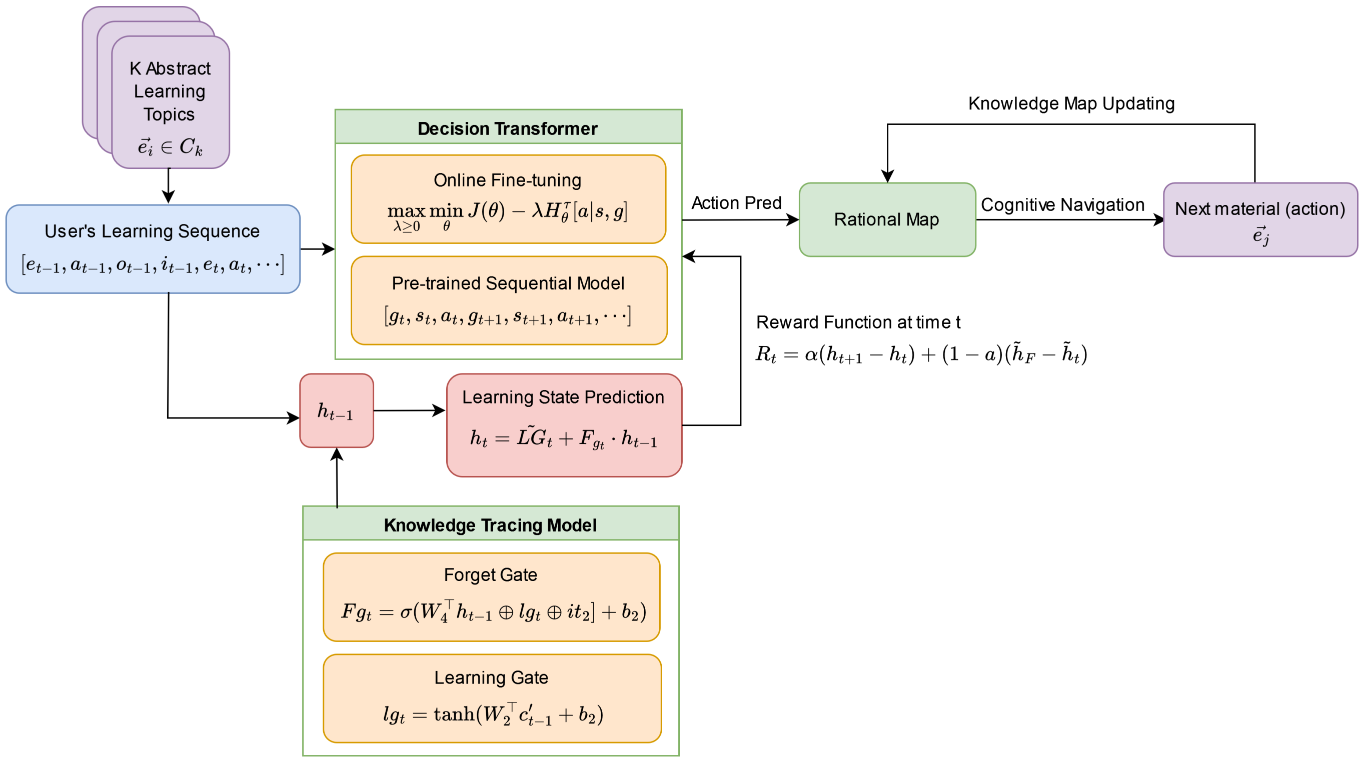
K Abstract Learning Topics
$\vec{e_i} \in C_k$
User's Learning Sequence
$[e_{t-1}, a_{t-1}, o_{t-1}, i_{t-1}, e_t, a_t, \cdots]$
Decision Transformer
Online Fine-tuning
$\max_{\lambda\geq 0}\min_{\theta} J(\theta) - \lambda H_\theta^\tau[a|s,g]$
Pre-trained Sequential Model
$[g_t, s_t, a_t, g_{t+1}, s_{t+1}, a_{t+1}, \cdots]$
Action Pred
Rational Map
Cognitive Navigation
Next material (action)
$\vec{e_j}$
Knowledge Map Updating
Reward Function at time t
$R_t = \alpha(h_{t+1} - h_t) + (1-a)(\tilde{h}_F - \tilde{h}_t)$
h_{t-1}
Learning State Prediction
$h_t = \tilde{LG}_t + F_{g_t} \cdot h_{t-1}$
Knowledge Tracing Model
Forget Gate
$Fg_t = \sigma(W_4^\top h_{t-1} \oplus lg_t \oplus it_2] + b_2)$
Learning Gate
$lg_t = \tanh(W_2^\top c'_{t-1} + b_2)$

Figure 3. The procedure of the learning path recommendation system