Exploring Multi-Objective Exercise Recommendations in Online Education Systems

Zhenya Huang[1], Qi Liu[1], Chengxiang Zhai[2,*], Yu Yin[1], Enhong Chen[1,*], Weibo Gao[1], Guoping Hu[3]
[1]Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology & School of Data Science, University of Science and Technology of China,
{huangzhy,yxonic}@mail.ustc.edu.cn; {qiliuql,cheneh}@ustc.edu.cn; iamwebgao@gmail.com
[2]University of Illinois at Urbana-Champaign, czhai@illinois.edu; [3]iFLYTEK Research, gphu@iflytek.com

Reporter: Zhenya Huang
Date: 2019.11.04

Outline

Background

- Online Education Systems become more and more popular
  - Abundant learning materials
    - E.g., exercise, course, video
  - Personalized learning service
    - Students can learn on their own pace
  - Various platforms
    - MOOC
    - Intelligent Tutoring System
    - Online Judging System

学堂在线

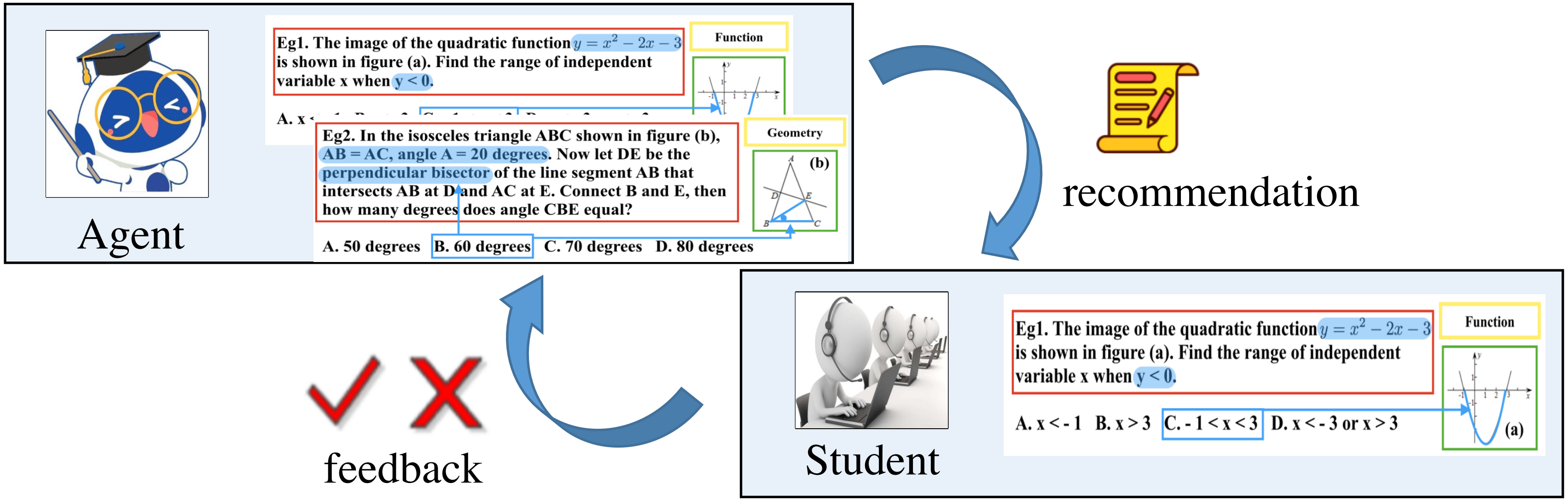
Khan Academy

智学网
zhixue.com

CODEFORCES $^{\beta}$
Sponsored by Telegram

Recommendation

- Recommender systems
  - Suggest suitable exercises instead of letting students self-seeking
  - Interactive systems between agent vs. student
- Key problem
  - Design an optimal strategy (algorithm) that can recommend the best exercise for each student at the right time

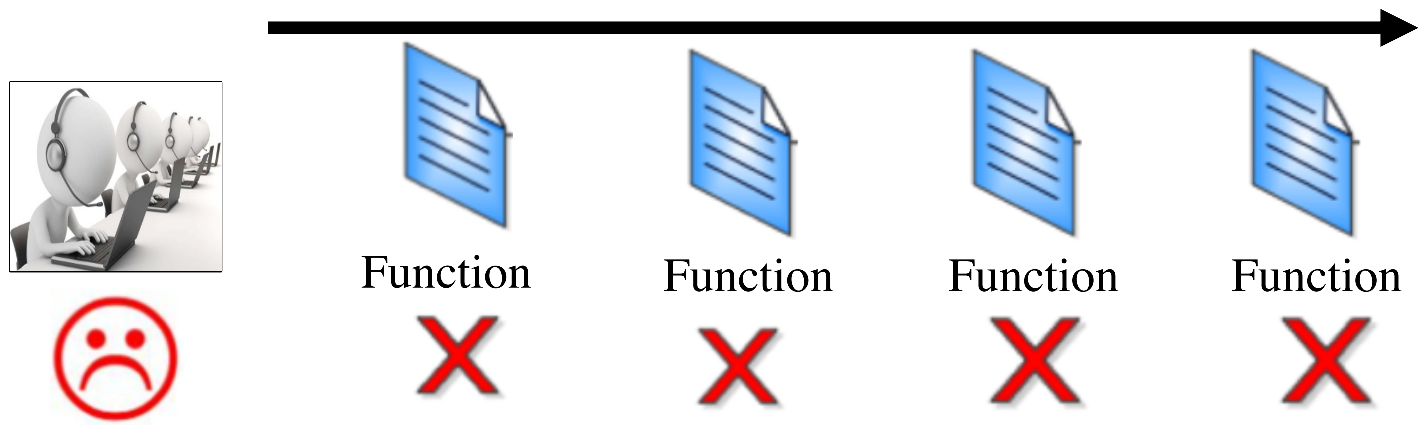


Related work

- Traditional recommendation for online learning
  - Basic idea:
    - Try to discover the weakness of students
    - Recommend the exercises that students may not learned well
- Existing methods
  - Educational psychology
    - Cognitive diagnosis studies
    - Traditional Q learning algorithm
  - Data-driven algorithm
    - Content-based methods
    - Collaborative filtering
    - Deep neural networks

Related work

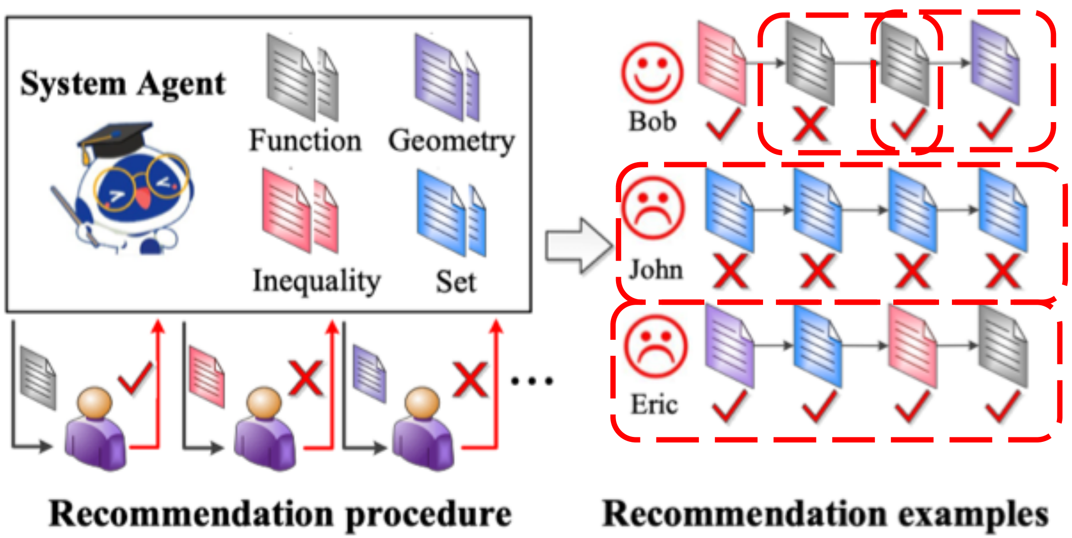
- Limitation
  - Single objective
    - Target at specific concepts with repeating exercising
  - Recommending non-mastered exercises
    - Always too hard
  - Student lose learning interests



What kinds of objectives should we concern in exercise recommendation?

Exercise Recommendation

- Multiple Objectives
  - Review & Explore
    - Review non-mastered concept vs. Seek new knowledge
  - Smoothness
    - Continuous recommendations on difficulty levels can not vary dramatically
  - Engagement
    - Keep learning
    - Some are challenging but some are "gifts"



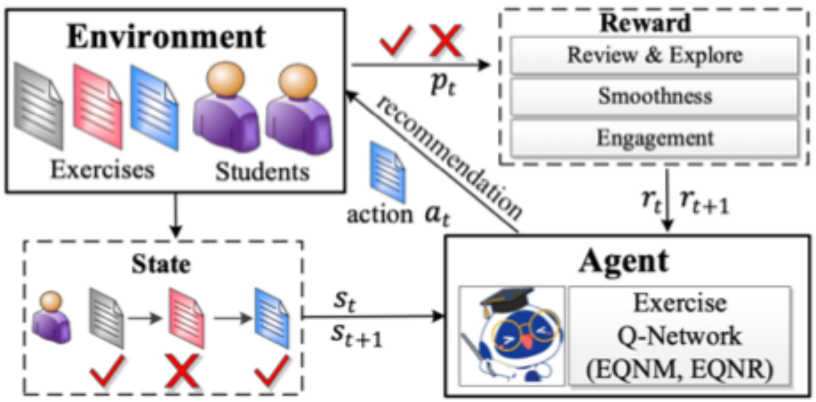
Exercise Recommendation

- Challenges
  - How to define multiple objectives?
    - Review & Explore
    - Smoothness
    - Engagement
  - How to enable flexible recommendations with considering above objectives simultaneously?
    - How to track students' learning states
    - How to quantify the objectives
  - Large space of exercise candidates

Outline

Problem Definition

- Given:
  - Student: exercising record $u = \{(e_1, p_1), (e_2, p_2), \cdots, (e_T, p_T)\}$,
  - Exercise: triplet $e = \{c, k, d\}$
    - Content: c is word sequence, $e = \{w_1, w_2, \ldots, w_M\}$
    - Knowledge (concept): $k \in K$(e.g., Function)
    - Difficulty level: d is the error rate, i.e., the percentage of students who answer exercise e wrong
- Markov Decision Process (MDP)
  - State s_t: the exercising history of the student
  - Action a_t: recommend an exercise e_{t+1} based on State s_t
  - Reward r(s_t, a_t): consider multiple objectives based on the performance feedback
  - Transition T: function: $S \times A \rightarrow S$, mapping state s_t to state s_{t+1}
- Goal:
  - Find an optimal policy π : S → A of recommending exercises to students, which maximizes the multi-objective rewards.



Outline

DRE framework

- At a glance
  - Deep reinforcement learning (Q-learning) framework
  - Exercise Q-network (EQN)
    - Estimate Q-values, generate exercise recommendation (taking action)
    - Track student learning states
    - Extract exercise semantics
    - Two Implementations
      - EQNM with Markov property
      - EQNR with Recurrent manner
  - Multi-objective Rewards
    - Review & Explore
    - Smoothness
    - Engagement
  - Off-policy training

DRE framework

- Optimization Objective
  - Future rewards R_t of state-action pair (s, a): $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$
  - Optimal action-value function

$$Q^*(s,a) = \mathbb{E}_{s'}[r + \gamma \max_{a'} Q^*(s', a')|s, a].$$

  - Compute the Q-values for all a′ ∈ A is infeasible
    - Estimate and store all state-action pairs (large exercise candidates)
    - Update all Q-values (student practices very few exercises)
  - Solution
    - Exercise Q-Network: as a network approximator θ

$$Q^*(s,a) \approx Q(s,a;\theta)$$

    - Minimize the objective function to estimate this network.

$$L_t(\theta_t) = \mathbb{E}_{s,a,r,s'}[(y - Q(s,a;\theta_t))^2],$$

$$y = \mathbb{E}_{s'}[r + \gamma \max_{a'} Q(s', a'; \theta_{t'})|s, a]$$

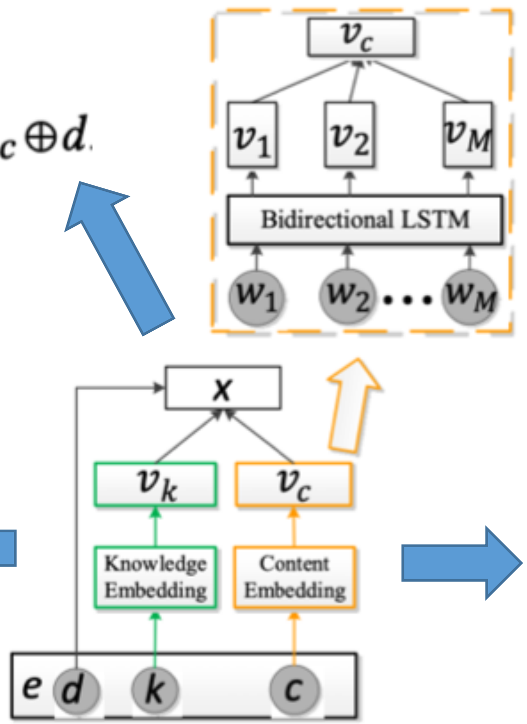
DRE framework

- Exercise Q-Network
  - Goal: estimate the action Q-value Q (s, a) of taking an action a at state s
    - Implement network approximator
  - Key points:
    - Learn the semantics of each exercise
      - Exercise Module
    - Learn the student knowledge states at each step
      - EQNM: Markov property
      - EQNR: Recurrent manner

Exercise Q-Network

- Exercise Module
  - Goal: learn the semantics of each exercise
  - Combination with knowledge, content and difficulty

$$x = v_k \oplus v_c \oplus d.$$



(c) Exercise Module

Knowledge embedding

$$v_k = \mathbf{W_k}^{\mathrm{T}} k.$$

Content embedding

$$\overrightarrow{v}_i = LSTM(w_i, \overrightarrow{v}_{i-1}; \theta_{\overrightarrow{v}}), \quad \overleftarrow{v}_i = LSTM(w_i, \overleftarrow{v}_{i-1}; \theta_{\overleftarrow{v}}),$$

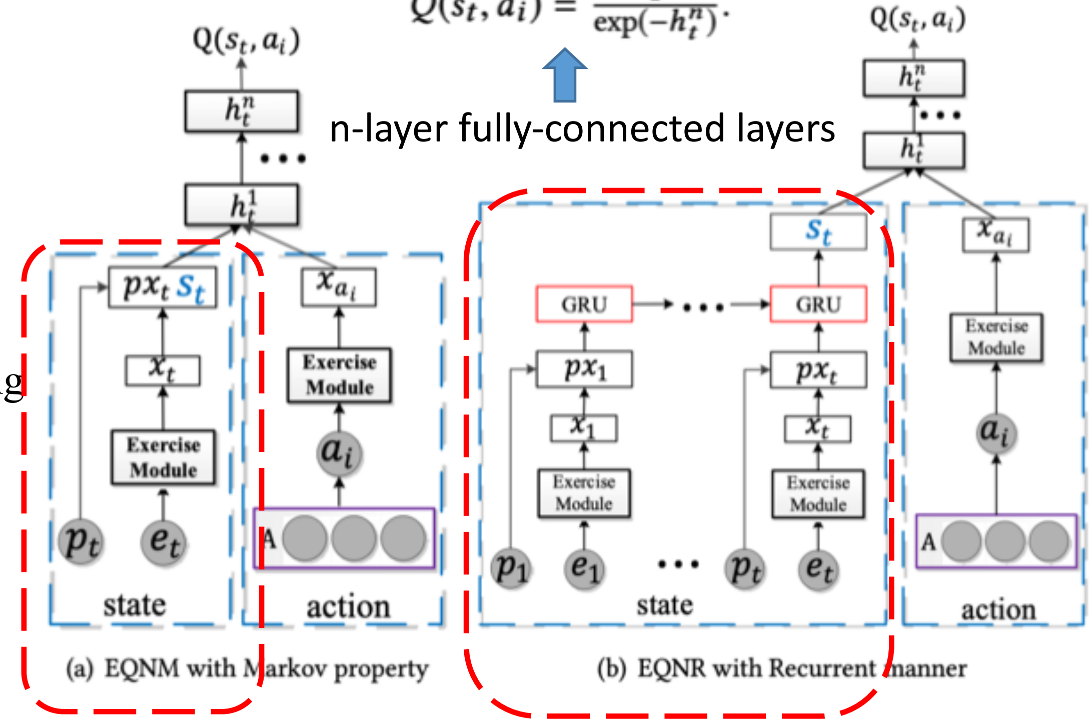
$$v_i = \overrightarrow{v}_i \oplus \overleftarrow{v}_i,$$

Exercise Q-Network

- Two implements
  - Goal: Learn the student knowledge states at each step
  - Estimate Q value Q(s, a): taking action at step t
    - EQNM: only observe current state $s_t = (e_t, p_t)$
    - EQNR: consider historical state trajectories: $s_t = \{(e_1, p_1), \cdots, (e_t, p_t)\}$

$$Q(s_t, a_i) = \frac{1}{\exp(-h_t^n)}.$$

Current state embedding



(a) EQNM with Markov property

(b) EQNR with Recurrent manner

$$l_j = GRU(px_j, l_{j-1}; \theta_g).$$

Multi-objective rewards

- Review & Explore
  - Intuition: review non-mastered concept vs. seek new knowledge
  - Review factor: review what they learned not well: punishment ($\beta_1 < 0$)
  - Explore factor: suggest to seek diverse concepts: stimulation ($\beta_2 > 0$)

$$r_1 = \begin{cases} \beta_1 & \boxed{\text{if} \quad p_t = 0 \quad \text{and} \quad k_{t+1} \cap k_t = \emptyset,} \\ \beta_2 & \boxed{\text{if} \quad k_{t+1} \setminus \{k_1 \cup k_2 \cup \cdots \cup k_t\} \neq \emptyset,} \\ 0 & \text{else.} \end{cases}$$

- Smoothness
  - Intuition: two continuous recommendations on difficulty levels should not vary dramatically
  - Negative squared loss

$$r_2 = \mathcal{L}(d_{t+1}, d_t) = -(d_{t+1} - d_t)^2,$$

Multi-objective rewards

- Engagement
  - Intuition: keep learning (interests), avoiding too hard or easy exercises all the time
  - Makes some recommendations are challenging but others seem “gifts”
    - Learning goal g
    - N historical performance φ on average

$$r_3 = 1 - |g - \varphi(u, N)|, \quad \varphi(u, N) = \frac{1}{N} \sum_{i=t-N}^{t} p_i,$$

- Balance multi-objective rewards

$$r = \alpha_1 \times r_1 + \alpha_2 \times r_2 + \alpha_3 \times r_3, \quad \{\alpha_1, \alpha_2, \alpha_3\} \in [0, 1].$$

Off-policy training

- Training with offline logs

Algorithm 1: DRE Learning with Off-Policy Training

Initialize replay memory $\mathcal{D}$ with capacity Z;
Initialize action-value function Q with random weights.;
for $u = 1, 2, \cdots, |U|$ **do**
  Randomly initialize state s_0;
  for $t = 1, 2, \cdots, T$ **do**
    Observe state $s_t = (e_t, p_t)$ in EQNM or $s_t = \{(e_1, p_1), \cdots, (e_t, p_t)\}$ in EQNR;
    Execute action a_t (e_{t+1}) from off-policy $\pi_o(s_t)$;
    Compute reward r_t according to p_{t+1} by Eq. (10);
    Set state $s_{t+1} = (e_{t+1}, p_{t+1})$ in EQNM or $s_{t+1} = \{(e_1, p_1), \cdots, (e_t, p_t), (e_{t+1}, p_{t+1})\}$ in EQNR;
    Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$;
    Sample minibatch of transition (s, a, r, s') from $\mathcal{D}$;
    $y = \begin{cases} r & \text{terminal } s' \\ r + \gamma \max_{a'}(Q(s', a'); \theta) & \text{non-terminal } s' \end{cases}$;
    Minimize $(y - Q(s, a); \theta)^2$ by Eq. (3);
  end
end

Learn from other agent policy

Experience reply

Two separate networks

Outline

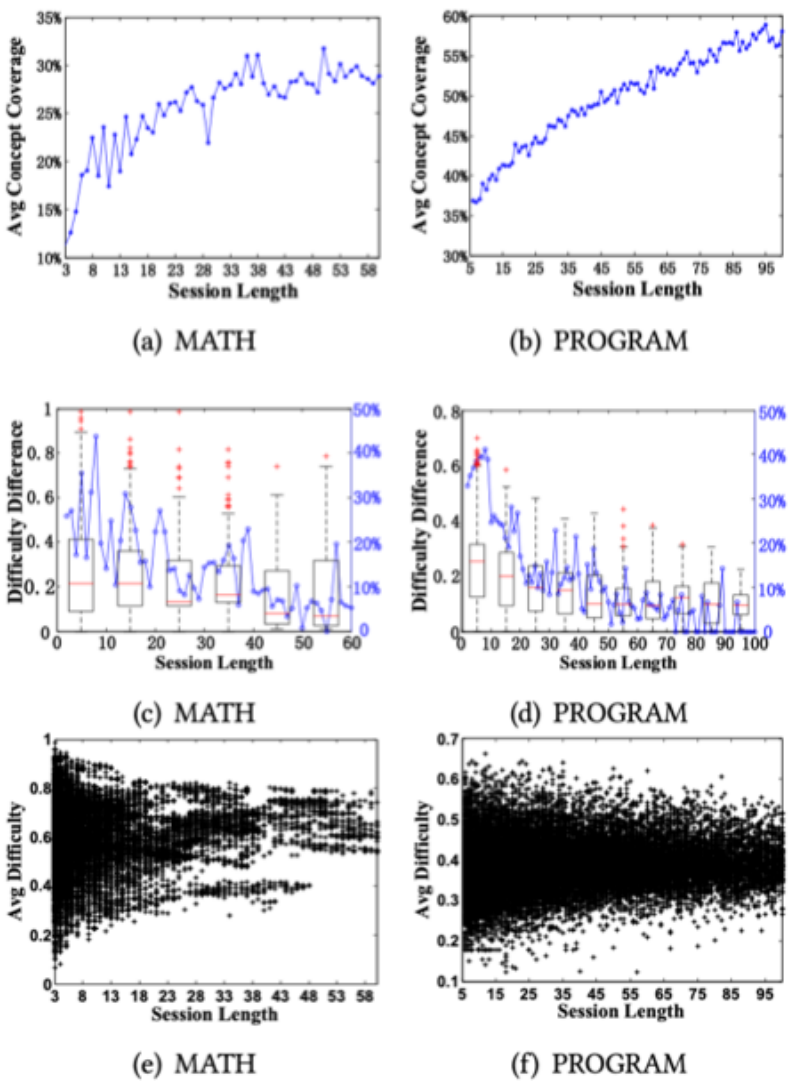
Experiment

- Datasets
  - MATH dataset (high school level)
  - PROGRAM dataset (oj platform)

Table 1: The statistics of the datasets.

Dataset	Num. Students	Num. Exercises	Num. Conceps	Num. records	Avg. records per student
MATH	52,010	2,464	37	1,272,264	24.5
PROGRAM	40,013	2,900	18	3,455,067	86.3

- Data analysis
  - Learning session
    - Interval timestamps last more than 24 (10) hours, split them into two sessions
  - Longer sessions have larger concept coverage
  - Longer sessions contain more samples with smaller difficulty differences
  - Longer sessions have exercises with medium difficulty on average
- https://base.ustc.edu.cn/data/DRE/



(a) MATH (b) PROGRAM
(c) MATH (d) PROGRAM
(e) MATH (f) PROGRAM

Experiment

- Offline Evaluation (Point-wise recommendation)
  - We evaluate methods on logged data
    - Static
    - Only contained pairs of student-exercise performance that had been recorded
    - Just know students' final scores on exercise
  - Ranking problem
    - For student: rank an exercise list at a particular time
    - Based on performance: from bad to good
  - Data partition: for each sequence, 70% training, 30% testing
  - DRE framework:

r (α_1=0, α_2=0, α_3=1) (Eq. (10)); r_3 (g=0, N=5) (Eq. (9))

$$r = \alpha_1 \times r_1 + \alpha_2 \times r_2 + \alpha_3 \times r_3, \quad \{\alpha_1, \alpha_2, \alpha_3\} \in [0,1]. \quad r_3 = 1 - |g - \varphi(u, N)|, \quad \varphi(u, N) = \frac{1}{N} \sum_{i=t-N}^{t} p_i,$$

  - Baseline:
    - Cognitive diagnosis: IRT
    - Recommender system: PMF, FM
    - Deep learning: DKT, DKVMN
    - Reinforcement learning: DQN

Experiment

- Offline Evaluation (Point-wise recommendation)

Table 2: The overall accuracy results of exercise recommendation in offline evaluation.

(a) **MATH**

Methods	NDCG@10	NDCG@15	MAP@10	MAP@15	F1@10	F1@15
IRT	0.5065	0.6235	0.3373	0.4463	0.2100	0.3464
PMF	0.4900	0.5986	0.3155	0.4163	0.2016	0.3347
FM	0.5123	0.6279	0.3419	0.4507	0.2123	0.3489
DKT	0.5587	0.7033	0.3959	0.5486	0.2797	0.4634
DKVMN	0.5657	0.7112	0.4021	0.5581	0.2895	0.4747
DQN	0.5031	0.7001	0.3191	0.5296	0.2912	0.5178
DREM	**0.6114**	0.7773	**0.4355**	0.6353	0.3559	0.6033
DRER	0.6129	**0.7813**	0.4337	**0.6435**	**0.3676**	**0.6099**

(b) **PROGRAM**

Methods	NDCG@10	NDCG@15	MAP@10	MAP@15	F1@10	F1@15
IRT	0.3369	0.4231	0.1852	0.2430	0.0879	0.1530
PMF	0.3330	0.4152	0.1810	0.2336	0.0842	0.1467
FM	0.3664	0.4456	0.2081	0.2617	0.0921	0.1567
DKT	0.3893	0.4924	0.2361	0.3197	0.1451	0.2445
DKVMN	0.3853	0.4889	0.2351	0.3226	0.1555	0.2620
DQN	0.3422	0.4901	0.1851	0.3095	0.1781	0.3266
DREM	0.4446	0.5638	0.2753	0.3834	0.1683	0.3325
DRER	**0.4538**	**0.5907**	**0.2802**	**0.4059**	**0.2091**	**0.3655**

- DRER and DREM generate accurate recommendations
- EQN > DQN: EQN well capture the state presentations of students
- DRER > DREM: EQNR can track the long-term dependency

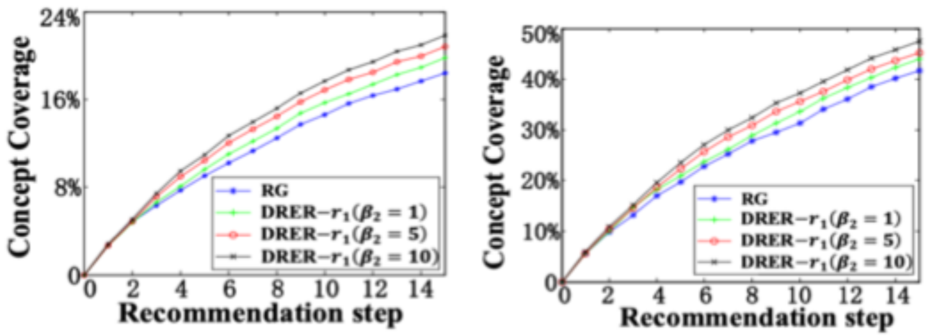
Experiment

- Online Evaluation (Sequence-wise recommendation)
  - We evaluate methods in a simulated environment
    - Implement a student simulator
    - Real-time interaction
  - Sequential recommendation scenario
    - For student: provide the best exercise step by step
    - Evaluate the effectiveness on three rewards (multiple objectives)
  - Preliminaries
    - Student simulator: EERNN (state-of-the-art)
    - Data partition: 50% for training simulator, 50% for training DRE framework

Experiment

- Online Evaluation (Sequence-wise recommendation)
  - Review & Explore

$$
r_1 = \begin{cases} \beta_1 & \text{if} \quad p_t = 0 \quad \text{and} \quad k_{t+1} \cap k_t = \emptyset, \\ \beta_2 & \text{if} \quad k_{t+1} \setminus \{k_1 \cup k_2 \cup \cdots \cup k_t\} \neq \emptyset, \\ 0 & \text{else.} \end{cases}
$$

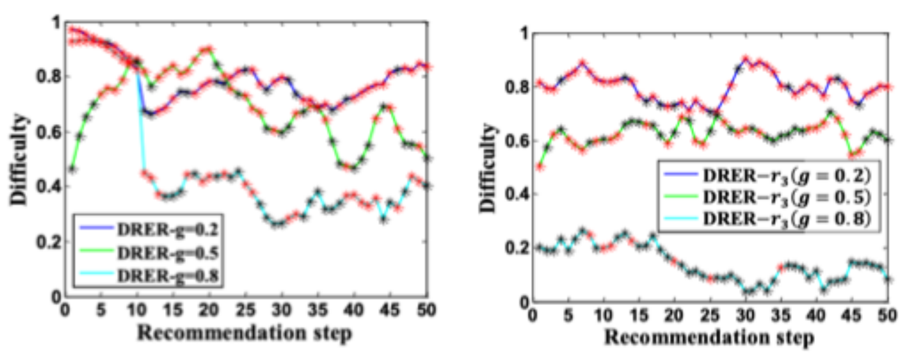


(a) MATH (b) PROGRAM

Figure 6: Results of Review & Explore reward.

- ✓ DRE with larger β_2 value has faster coverage growth speed

  - Smoothness vs. Engagement



(a) MATH (b) PROGRAM

Figure 7: Results of Smoothness vs. Engagement rewards.

- ✓ The difficulty levels of recommendations do not vary dramatically in most cases
- ✓ If we set learning goal g with lower value (0.2), DRE would recommend more difficult exercises

Outline

Experiment

- Conclusion
  - Deep Reinforcement learning framework for Exercise recommendation
  - Two Exercise Q-Networks (EQN) to select exercise recommendations following different mechanisms (Markov, Recurrent)
  - Design three domain-specific rewards to find the optimal recommendation strategy
    - Review & Explore, Smoothness and Engagement

- Future work
  - Seek more ways to learn the reward settings automatically
    - Behaviors: if the student solves exercises very quickly, set g with a lower value
  - Develop a system and apply DRE framework online
    - Get and test real-world feedback
    - Find more direct method to evaluate the students' satisfaction.
  - Extend to more general domains
    - Online shopping, e-commerce, POI service etc

Thanks for your listening!

huangzhy@mail.ustc.edu.cn

Welcome to our poster for more details tonight