A DATA SHEET

We follow the documentation frameworks provided by Gebru et al. [12].

A.1 Motivation

For what purpose was the dataset created? While recently many works claim that they successfully apply reinforcement learning algorithms to recommendation tasks, the experimental settings in these works (*i*) lacks datasets collected from real-world applications, and only test on artificial datasets and semi-simulated datasets with unreasonable data transformations; (*ii*) often tests in the environment where the RL agent trains on, and lacks uniform and unbiased evaluation criteria. This motivates us to build the first real-world dataset, provide all details from raw data to environment construction, propose a unified and principled evaluation framework, and evaluate to which extent the state-of-the-art models have progressed so far in terms of RL-based RS.

Who created the dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)? Fuxi AI Lab, the Netease Inc.

A.2 Composition:

What do the instances that comprise the dataset represent (e.g., documents, photos, people, countries)? Are there multiple types of instances (e.g., movies, users, and ratings; people and interactions between them; nodes and edges)? RL4RS contains item metadata and user feedbacks in the dataset.

How many instances are there in total (of each type, if appropriate)? There are 1719316 user sessions for RL4RS-Slate dataset and 958566 user sessions for RL4RS-SeqSlate dataset.

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? If the dataset is a sample, then what is the larger set? Is the sample representative of the larger set (e.g., geographic coverage)? If so, please describe how this representativeness was validated/verified. If it is not representative of the larger set, please describe why not (e.g., to cover a more diverse range of instances, because instances were withheld or unavailable). It is collected from an online application in a week without sampling. It is representative.

Are relationships between individual instances made explicitly (e.g., users' movie ratings, social network links)? If so, please describe how these relationships are made explicit. N/A.

Are there recommended data splits (e.g., training, development/validation, testing)? If so, please provide a description of these splits, explaining the rationale behind them. The code of data splits is provided in https://github.com/fuxiAIlab/RL4RS/blob/main/reproductions/run_split.sh.

Are there any errors, sources of noise, or redundancies in the dataset? If so, please provide a description. No.

Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)? If it links to or relies on external resources, a) are there guarantees that they will exist, and remain constant, over time; b) are there official archival versions of the complete dataset (i.e., including the external resources as they existed at the time the dataset was created); c) are there any restrictions] (e.g., licenses, fees) associated with any of the external resources that might apply to a future user? Please provide descriptions of all external resources and any restrictions associated with them, as well as links or other access points, as appropriate. RL4RS is self-contained.

Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctorpatient confidentiality, data that includes the content of individuals' non-public communications)? If so, please provide a description. No, all the samples are public available.

Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? If so, please describe why. No.

Does the dataset relate to people? If not, you may skip the remaining questions in this section. No.

A.3 Uses

Has the dataset been used for any tasks already? If so, please provide a description? It is proposed to use for reinforcement learning based recommendation task.

Is there a repository that links to any or all papers or systems that use the dataset? If so, please provide a link or other access point. It is a new dataset. A older and smaller dataset collected from the same application is used in a kaggle competition https://www.kaggle.com/c/bigdata2021-rl-recsys/overview.

What (other) tasks could the dataset be used for? Many other tasks like neural combinatorial optimization can be also used.

Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses? For example, is there anything that a future user might need to know to avoid uses that could result in unfair treatment of individuals or groups (e.g., stereotyping, quality of service issues) or other undesirable harms (e.g., financial harms, legal risks) If so, please provide a description. Is there anything a future user could do to mitigate these undesirable harms? N/A

Are there tasks for which the dataset should not be used? If so, please provide a description. N/A

A.4 Distribution

Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created? All datasets are released to the public.

How will the dataset will be distributed (e.g., tarball on website, API, GitHub)? All datasets are released on Zenodo https://zenodo.org/record/6622390#.YqBBpRNBxQK and our github repository https://github.com/fuxiAIlab/RL4RS. The dataset DOI is 10.5281/zenodo.6622390.

When will the dataset be distributed? It has been released now.

Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use (ToU)?. Our dataset is distributed under the CC BY-SA 4.0 license, see https://github.com/fuxiAIlab/RL4RS/blob/main/LICENSE.

Have any third parties imposed IP-based or other restrictions on the data associated with the instances? If so, please describe these restrictions, and provide a link or other access point to, or otherwise reproduce, any relevant licensing terms, as well as any fees associated with these restrictions. No.

Do any export controls or other regulatory restrictions apply to the dataset or to individual instances? If so, please describe these restrictions, and provide a link or other access point to, or otherwise reproduce, any supporting documentation. No.

A.5 Maintenance

How can the owner/curator/manager of the dataset be contacted (e.g., email address)? Kai Wang (`wangkai02@corp.netease.com`) and Runze Wu (`wurunze1@corp.netease.com`) will be responsible for maintenance.

Will the dataset be updated (e.g., to correct labeling errors, add new instances, delete instances)? Yes. If we include more tasks or find any errors, we will correct the dataset and update the leaderboard accordingly. It will be updated on our website.

If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so? They can contact us via email for the contribution.

B WORKING WITH RL-BASED RS DATA USING RL4RS

In order to facilitate the entry of new practitioners to the field of RL-based recommender system, we provide some detailed guidelines for working with the datasets we provide and for managing new ones in the framework of the `rl4rs` package. We have provided some tools required for RL-based RS task (e.g., data understanding tools), and will be continuing to develop and support more functionalities.

B.1 Assembling New Datasets

Data Characteristic Examine. Recommender systems are an area that relies heavily on data. When appling RL into RS, one big challenge is problem formulation. It is easy to accidentally prepare data that does not conform well to the MDP definition, and applying RL on ill-formulated problems isa costly process. Here, we develop a data understanding tool. Using a data-driven method together with heuristics, the tool checks whether the properties of RS datasets conform to the RLframework. RL4RS provides three usecases for RS datasets to help users quickly use the tool to check their new datasets. In practice, it may help us understand the dataset in early stages, catch improperly defined RL problems, and lead to a better prob-lem definition (e.g., reward definition)

Batched Environment Preparation. Different from gym environments, simulation environments of RL-based RS tasks are often based on a pre-trained neural network. In most recommended scenarios, this often includes several thousand dimensional features, millions of samples, several GB files, and tens of milliseconds of inference delay. These characteristics make this environment difficult to be accelerated by the existing RL libraries, and lead to the model can not be trained in an acceptable time. Based on the popular RLLib library, RL4RS developes two solutions: file-based RL environment and HTTP-based vector environment. The former enables random sampling and sequential access to datasets exceeding memory size, and can be easy to extend to distributed file systems. The latter allows the environment processes batch data at one time and to be deployed on multiple servers to accelerate the generation of samples. Users only need to process new dataset into the specified data format to use these two functionalities.

B.2 Developing and Benchmarking New Algorithms

Comparing Algorithms. The best practices for RL-based RS are still poorly explored. Not only do we compare different RL models on a given environment model, but we also need to compare different environment construction methods, different batch RL methods which learn directly from offline data, and even different MDP definitions. Just for comparing RL, we need to consider whether actions are defined as continuous or discrete, use hidden layer or raw feature as observation, use action mask as hard or soft constraint, etc. RL4RS suits provide all option combinations to help users to explore the best practices for their new dataset.

Evaluation Outside the Environment. Predictions can be tested with various metrics. Current RL evaluation methodology, i.e., training and evaluating on the same environment, is not yet well-suited for RL-based RS tasks. RL4RS provides some other different evaluation methods, including evaluation of simulation environments, offline policy evaluation and evaluating on the environment built from test set. In the future, we will add more evaluation metrics. To calculate new metrics for older models, when benchmarking an algorithm, specific predictions should be stored and not only metrics. In RL4RS, we provide standardized evaluation classes to calculate the appropriate metrics from the outputs of a given algorithm. Users only need to implement their custom environment class for the new dataset by inheriting the base class.

C EXPERIMENTAL DETAILS

C.1 Implementation Details of Data Preprocess

Given the raw logged data described in Section 3, specifically, we transforms the logged data collected in the following row format:

- **MDP ID**: A unique ID for the Markov Decision Process (MDP) chain that this training example is a part of.
- **Sequence ID**: A number representing the location of the state in the MDP (i.e. a timestamp).
- **State**: The features of current step that consists of user features and context features.
- **Observation**: The 256-dim hidden layer embedding of the pre-trained environment model with user features and context features of current step as input. Or we can use the raw state features as observation.

- **Action**: The item ID taken at the current step. Or we can use item embeddings can be used to represent item for continuous environment setting.
- **Action Features**: The features of item taken at the current step, including item embedding.
- **Action Probability**: The probability that the behavior policy took the action.
- **Action Mask**: An list of possible items at the current step.
- **Reward**: The reward of the current step.
- **Next State**: The state features after acting the logged action.
- **Next Observation**: The 256-dim hidden layer embedding of next state features.
- **Next Action**: The item ID recommended at the next step.
- **Next Action Features**: The features of item recommended at the next step.
- **Next Action Probability**: The probability of the item that is recommended at the next step.
- **Next Action Mask**: A list of items that are allowed to recommend at the next step.
- **Terminal**: A 0/1 number representing whether it is the last state.

The details of data splits and the above data transformations are provided in the open-source code.

C.2 Implementation Details of Simulation Environment Construction

The detailed network architectures of baselines are provided in https://github.com/fuxiAIlab/RL4RS/tree/main/rl4rs/nets. Here we introduce the three tasks for simulation environment construction.

Slate-wise Classification Task. The question can be transformed into a multi-class classification task, where the feedbacks of users to the nine items in the page are considered as a whole. Remove impossible situations, there are 22 possible classes. Given the nine items in the page $\langle a_1, a_2, \ldots, a_9 \rangle$, corresponding user feedbacks $\langle o_1, o_2, \ldots, o_9 \rangle$, and the model 0/1 prediction $\langle \hat{o}_1, \hat{o}_2, \ldots, \hat{o}_9 \rangle$, the accuracy metric is defined as the identical accuracy between $\langle o_1, o_2, \ldots, o_9 \rangle$ and $\langle \hat{o}_1, \hat{o}_2, \ldots, \hat{o}_9 \rangle$.

Item-wise Rank Task. The question can be transformed into a item ranking task (or a multi-label task), where the feedbacks of users to the nine items in the page are treated as different labels. Given the nine items in the page $\langle a_1, a_2, \ldots, a_9 \rangle$, corresponding user feedbacks $\langle o_1, o_2, \ldots, o_9 \rangle$, and the user's purchase probability prediction of the model $\langle \hat{y}_1, \hat{y}_2, \ldots, \hat{y}_9 \rangle$, the AUC, precision and recall metrics can be calculated.

Item-wise Classification Task. The question can be transformed into a binary classification task, where the feedbacks of users to the nine items in the page are considered item by item. Given the nine items in the page as recommendation context, the model outputs the user's purchase probability $\hat{y}_j$ for item j. The AUC and accuracy metrics are calculated based on the purchase probability y and ground-truth label o.

Table 10: The fine-tuning details of model-free reinforcement learning algorithms under the setting of discrete action space.

	Exploration	Training Config	Model Config
PG	SoftQ	steps:6400000 batch_size:256 hidden_units:128 learning_rate: 0.0004	default
DQN	SoftQ	steps:6400000 batch_size:256 hidden_units:128 learning_rate: 0.0001	target_update_freq:200 double_q:True dueling:False buffer_size: 100000 n_step: 1
RAINBOW	SoftQ	steps:6400000 batch_size:256 learning_rate: 0.0001 no custom encoder	target_update_freq:200 double_q:True dueling:True buffer_size: 100000 n_step: 3 num_atoms:8
A2C	SoftQ	steps:6400000 batch_size:256 hidden_units:128 learning_rate: 0.0001	use_critic:True use_gae:True vf_loss_coeff:0.5 entropy_coeff:0.01 grad_clip:10
PPO	SoftQ	steps:6400000 batch_size:256 hidden_units:128 learning_rate: 0.0001	use_critic:True use_gae:True kl_coeff:0.2 vf_loss_coeff:0.5 clip_param:0.3 kl_target:0.1 sgd_minibatch_size:256 num_sgd_iter:1
Exact-k	EpsilonGreedy	steps:5120000 batch_size:512 hidden_units:256 actor_lr: 0.001 critic_lr: 0.005	num_heads:4 num_layers:1 num_blocks:2 use_mha:True

C.3 Fine-tuning Details of Model-free Reinforcement Learning

In RL4RS, model-free RL algorithms are implemented based on the popular open-source library RLLib [25]. The gamma is set to 1. The fine-tuned models and reproduction scripts are all included in the Github repository. All baselines share the same encoder network architecture. The detailed parameters are shown in Table 11.

C.4 Fine-tuning Details of Batch Reinforcement Learning

In RL4RS, batch RL algorithms are implemented based on the open-source batch RL library d3rlpy [33]. The gamma is set to 1. The fine-tuned models and reproduction scripts are all included in the Github repository. All baselines share the same encoder network architecture. The detailed parameters are shown in Table 12.

D DATA UNDERSTANDING

One big challenge of applied RL is problem formulation. Traditional recommendation scenarios and datasets are not always suitable for modeling as MDP problems where some sort of long-term reward is optimized in a sequential setting. It is easy to accidentally prepare data that does not conform well to the MDP definition, and applying RL on ill-formulated problems is a costly process. Here, we develop a data understanding tool. Using a data-driven method together with heuristics, the tool checks whether the properties of RS datasets conform to the RL framework.

Table 11: The fine-tuning details of model-free reinforcement learning algorithms under the setting of continuous action space.

	Exploration	Training Config	Model Config
PG-conti	StochasticSampling	steps:6400000 batch_size:256 hidden_units:128 learning_rate: 0.0004 action_emb_size:32	default
A2C-conti	StochasticSampling	steps:6400000 batch_size:256 hidden_units:128 learning_rate: 0.0001 action_emb_size:32	use_critic:True use_gae:True vf_loss_coeff:0.5 entropy_coeff:0.01 grad_clip:10
PPO-conti	StochasticSampling	steps:6400000 batch_size:256 hidden_units:128 learning_rate: 0.0001 action_emb_size:32	use_critic:True use_gae:True kl_coeff:0.2 vf_loss_coeff:0.5 clip_param:0.3 kl_target:0.1 sgd_minibatch_size:256 num_sgd_iter:1
DDPG	OrnsteinUhlenbeckNoise	steps:6400000 batch_size:256 hidden_units:128 learning_rate: 0.001 action_emb_size:32	twin_q:False target_noise:0.2
TD3	OrnsteinUhlenbeckNoise	steps:6400000 batch_size:256 hidden_units:128 learning_rate: 0.001 action_emb_size:32	random_timesteps:10000

Table 12: The fine-tuning details of batch reinforcement learning algorithms.

	Training Config	Model Config
BC	epoch:10 batch_size:256 hidden_units:256 learning_rate: 0.0001	bc_beta:0 reward_scaler:'None'
BCQ	epoch:10 batch_size:256 hidden_units:256 learning_rate: 6.25e-5	action_flexibility:0.3 bcq_beta:0.5 share_encoder:False reward_scaler:'None' target_update_interval:8000
CQL	epoch:10 batch_size:256 hidden_units:256 learning_rate: 6.25e-5	gamma:1.0 share_encoder:True reward_scaler:'standard' cql_alpha:1
MOPO	epoch:10 batch_size:256 hidden_units:256 learning_rate: 3e-4	n_critics:2 rollout_horizon:5 obs_scaler:'min_max' reward_scaler:'standard'
COMBO	epoch:10 batch_size:256 hidden_units:256 learning_rate: 3e-4	conservative_weight:1.0 n_critics:2 obs_scaler:'min_max' reward_scaler:'standard' n_action_samples:10

D.1 Long-term Impact

An optimal policy should take into account the long-term impact of a recommendation on the user's future choices. To maximize the accumulative rewards, we might suggest an item whose immediate reward is low but leads to more likely or more profitable rewards in the future. For example, when the products sold are books, by recommending a book for which there is a sequel, we may increase the likelihood that this sequel will be purchased later. Heuristically, the way to measure whether a recommendation problem should be modeled as an RL problem is to see whether a recommendation decision has a long-term impact. In terms of reinforcement learning formula, the recommendation of step t is to maximize $r(s_t, a_t) + V^*(s_{t+1})$, where $r(s_t, a_t)$ is the expected reward when recommending item a_t at state s_t, and $V^*(s_{t+1})$ represents the maximum reward of next state under current policy. Since there is always a large state value to represent users retention, we further denote $A(s_t, a_t) = V^*(s_{t+1}) - V^*(s_t, \cdot)$ as the difference (advantage) between the future reward of performing a_t and the averaged future rewards, i.e., the long-term impact of performing a_t at state s_t. When there is no long-term impact or the long-term impact is small, the RL problem degenerates into an ordinary sequential recommendation problem, which only maximizes the reward of the current step.

Though we can estimate the long-term impact by training an RL model, it is too complex. Instead, we establish a data understanding tool to quantify the long-term impact without the requirement of establishing environment models or learning a value function. This tool is easy to use by simplifying RL as sequence modeling problems and expected rewards as decoded sequence scores. Similar ideas have been developed in Trajectory Transformer [22] and Decision Transformer [2], in which states, actions, and returns are fed into a GPT [29] architecture and actions are decoded autoregressively.

First, the tool fits a Transformer-based sequence-to-sequence model [39] on each offline RS dataset. It encodes the user context features and decodes K items that users may click or buy, which means that the recommendation system will recommend these K items in turn in the next K steps (considering that most RS datasets do not provide a complete page context, here, only one item is recommended at a time, eliminating the slate effect between items within one page). We consider using the decode sequence generated by greedy search to represent greedy recommendation (SL-based strategy), and the sequence generated by beam search to represent the recommendation result generated by the optimal RL policy. We use beam-search width of 100. When there is a significant long-term item impact in the dataset, the items recommended in the previous steps may not have high immediate impacts, but their long-term impacts are large. That is, the immediate reward of the first k item in the best decode sequence may account for a small proportion in the final score of the sequence (experiment I). On the other hand, we can compare the sequences score of greedy strategy (even the sequence composed of only hot items which have high immediate reward) with the score of optimal sequence to check whether the greedy strategy is good enough (experiment II).

D.2 Experiment

We build experiments on the following datasets, MovieLens-25m, RecSys15 YooChoose, and RL4RS-Slate (has the same result as RL4RS-SeqSlate under this experiment setting). Without losing generality, we only consider the long-term impact within 5 steps. For each dataset, we choose 10000 users randomly as test set, and others as train set. For each user, we calculate the greedy search

Table 13: The comparison of item's long-term impact between different datasets.

Score. of	RecSys15	MovieLens	RL4RS-Slate
1-Pearson	0.11	0.13	0.03
1-Spearman	0.10	0.11	0.02
2-Pearson	0.16	0.17	0.06
2-Spearman	0.15	0.16	0.03
3-Pearson	0.24	0.25	0.14
3-Spearman	0.23	0.23	0.08
4-Pearson	0.39	0.38	0.36
4-Spearman	0.39	0.36	0.33
5-Pearson	1.00	1.00	1.00
5-Spearman	1.00	1.00	1.00

Table 14: The comparison of decode sequence scores between different datasets.

Score. of	5%	20%	greedy.	hot 5%	hot 20%
RecSys15	1.00	0.53	1.26	0.81	0.42
MovieLens	1.00	0.64	0.99	0.98	0.62
Last.fm	1.00	0.47	1.09	0.90	0.42
CIKMCup2016	1.00	0.30	4.50	0.99	0.28
RL4RS-Slate	1.00	0.76	0.62	0.01	0.01

result and the top 100 item sequences (beam search width as 100). In the first experiment, we report the Pearson Correlation Coefficient and Spearman Rank Correlation Coefficient between the scores of the first k items (immediate reward) and the final sequence scores, denoted as k-Pearson and k-Spearman respectively. If the item's long-term impact is significant, the coefficients should be small, i.e., the immediate reward of the first item in the decode sequence accounts for a small proportion of the final score of the sequence. We report the results on the top 100 item sequences generated by beam search, as shown in Table 13. It can be seen that RecSys15 and MovieLens show positive correlations at the beginning. RL4RS dataset achieves a significantly lower spearman rank coefficient at the first one and two items than the other two datasets. It means, in RL4RS dataset, the first k items recommended do not have high immediate impacts, but have great long-term impacts. Although the first experiment can not absolutely indicate whether a dataset is suitable for reinforcement learning, it provides a tool for comparison of the degree of long-term impact characteristics between different datasets.

In the second experiment, we aim to distinguish the dataset property by checking whether the greedy strategy is already good enough. We report the averaged score of the top 5 (5% quantile) decode sequence, the top 20 (20% quantile) sequence, and the score of sequence generated by greedy strategy. We also calculate the averaged score of the first 5% quantile and 20% quantile sequences when limiting the candidate items to hot items (100 items with the highest probability at the first decode step). All these scores are normalized by the average score of the top 5 sequences for each dataset. The results are shown in Table 14. It can be seen that there is a significant gap in the scores between the best 5% item sequences and greedy strategy for RL4RS dataset, let alone the sequences composed of hot items. It shows that it is necessary to model the RL4RS dataset as an RL problem. For traditional RS datasets, on the contrary, the score of greedy strategy is very close to that of the best RL policy (top 5 decode sequence). It indicates that the greedy strategy (i.e., sequential RS methods) is well enough to model these scenarios.

Although the data understanding tool is based on heuristics, in practice, it may help us understand the dataset in the early stages, catch improperly defined RL problems, and lead to a better problem definition (e.g., reward definition).

E POLICY EVALUATION FRAMEWORK

In the previous section, we provide a simple performance comparison between model-free RL and batch RL. These results are collected following the traditional RL evaluation framework, that is, training and evaluating on the same environment. However, in real applications, it is usually rare to have access to a perfect simulator. A policy should be well evaluated before deployment because policy deployment affects the real world and can be costly. In applied settings, policy evaluation is not a simple task that can be quantified with only a single indicator (e.g., reward). As shown in Figure 6, we attempt to propose a new evaluation framework for RL-based RS, including environment simulation evaluation, counterfactual policy evaluation (offline policy evaluation), and evaluation on simulation environments built from test set (online policy evaluation).

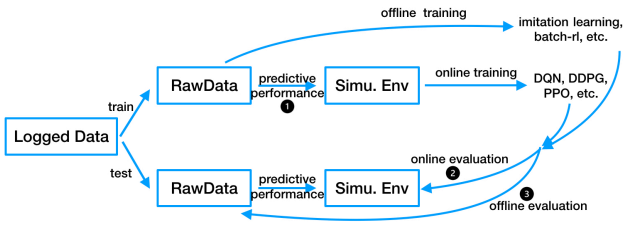

Figure 6: The evaluation framework for the RL-based RS task.

E.1 Evaluation On Another Simulation Environment

A possible evaluation method is to establish an environment on the test set to evaluate the algorithm. Different from testing in the same environment, we emphasize the establishment of environment model on test set according to the evaluation framework of supervised learning. In many recommendation scenarios, the training set and test set need to be divided according to time, which makes this more important. Another popular evaluation method adopted by most current works is to divide users before policy training. Although the training and test data are not overlapped, they are sharing the same environment. This method still exists a risk of over-fitting, because the estimated reward of the test set will be affected by the train set through the shared environment.

According to the method described in Section 5.1, we construct the environment model on the training set and the test set respectively. We use the most strict dataset dividing setting, that is, Dataset-SL as the train set and Dataset-RL as the test set. Not

Table 16: Performance comparison between batch RL algorithms on different experiment settings (setting 1, 2, 3, and 4 represent the setting of building environment and training on all datasets then test on Dataset-RL, building environment on all datasets and training on Dataset-SL then test on Dataset-RL, building environment and training on Dataset-SL then test on Dataset-RL, and building environment and training on Dataset-SL then test on the environment built from Dataset-RL respectively). On-going experiments.

	RL4RS-Slate			RL4RS-SeqSlate		
	BC	BCQ	CQL	BC	BCQ	CQL
Setting 1	98.0	132.9	107.2	240.7	277.2	259.4
Setting 2	98.8	133.4	98.8	244.7	273.7	268.3
Setting 3	103.7	200.7	109.3	230.6	288.9	274.2
Setting 4	107.1	133.6	108.6	139.5	196.6	174.6

Table 15: Reward estimation performance comparison of DIEN-based environment simulator between different environment simulation settings (setting 1, 2, 3, and 4 represent the setting of training on all dataset then testing on Dataset-SL, training on all dataset then testing on Dataset-RL, training on Dataset-RL then testing on Dataset-SL, and training on Dataset-SL then testing on Dataset-RL respectively). On-going experiments.

	RL4RS-Slate			RL4RS-SeqSlate		
	mean.	abs.	std.	mean	abs.	std.
Setting 1	-2.3	38.1	66.5	-13.3	63.5	99.3
Setting 2	-2.7	36.3	61.4	-6.8	65.0	97.7
Setting 3	1.2	42.4	69.4	2.4	71.5	106.5
Setting 4	4.5	40.6	64.4	23.3	81.0	106.9

only they are not overlapped on time, but also there are considerable differences in data distribution between them. For model-free RL and batch RL, we compare the performance of the following three settings: (1) constructing environment model on Dataset-SL plus Dataset-RL, training the policy on Dataset-SL, evaluating on Dataset-SL; (2) constructing environment model on Dataset-SL, training the policy on Dataset-SL, evaluating on Dataset-RL; (3) constructing environment model on Dataset-SL, training the policy on Dataset-SL, evaluating on the environment model built from Dataset-RL. For environment simulation, we provide the result of the following two settings: (1) constructing environment model on Dataset-SL, and then evaluating on Dataset-RL; and (2) constructing environment model on Dataset-RL, and then evaluating on Dataset-SL; The results of training on all dataset and evaluate on Dataset-SL and Dataset-RL are also provided as a baseline. The results are as follows.

From Table 15, it can be found that comparing setting 3/4 and setting 1/2, there is basically a 10% increase in the absolute mean indicator and a 5% increase in the standard deviation indicator on RL4RS-Slate dataset. On RL4RS-SeqSlate dataset, the indicator difference widens further, especially with setting 4 versus setting 2. Since the initial error of environment construction is already large, the impact of dataset division is difficult to measure directly. It needs to be further revealed by the results of the RL algorithms.

Table 16 shows the results of the three batch RL algorithms under the three experimental settings mentioned above and the baseline experimental settings (setting 1). Because the experiments are time-consuming, only a few rounds of experiments have been done at present. We will provide the averaged results of more rounds in the future. From the comparison between setting 3/4 and setting 2, the choice of environment construction dataset has a great impact on the final result. Compared with setting 2 and setting 1, the results do not show much difference (except CQL algorithm). It may be because they both use all dataset to construct the environment model. Although setting 2 only trains on Dataset-SL, it also learns a lot from Dataset-RL.