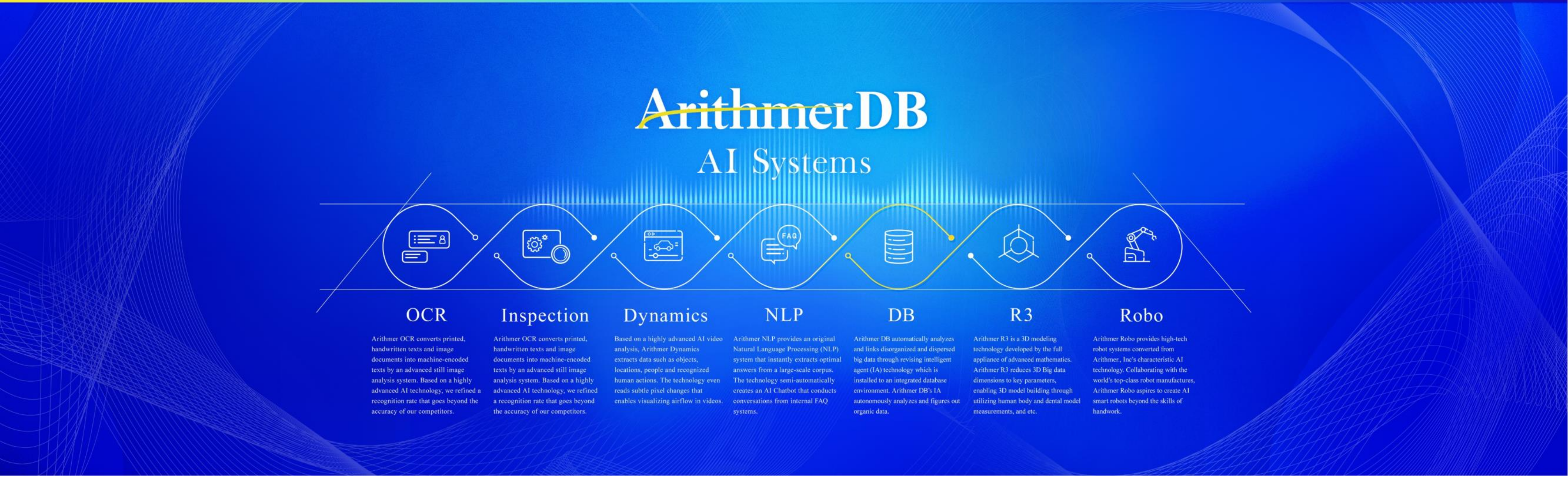


Recommendation Algorithm Using Reinforcement Learning

Arithmer DB Lu Juanjuan

2020/09/15

Self-Introduction

- Lu Juanjuan
  - Graduated School
    - Tokyo Institute of Technology
      - Ishida Takashi Laboratory, Department of Computer Science , School of Computing
        Master research domain:
        Drug discovery by applying machine learning technologies
  - Current Job
    - Arithmer Inc. （Home page: https://arithmer.co.jp/en/)
      - Application of Machine Learning/ Data Analysis

Outline

1. Background

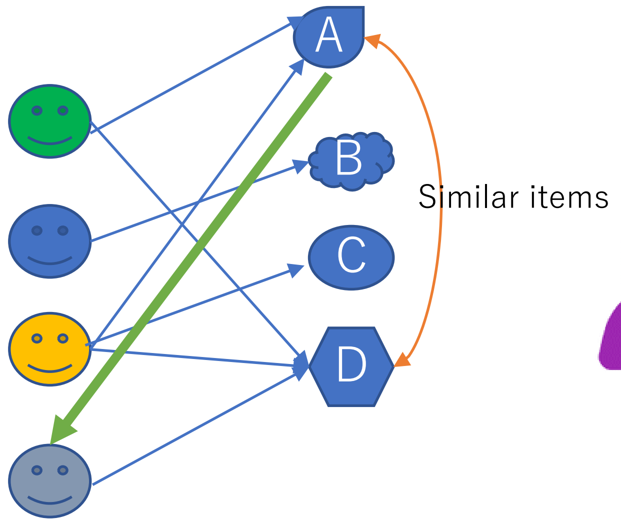
1. Recommendation System
2. Reinforcement Learning
3. Recommendation System using Reinforcement Learning

2. System Structure

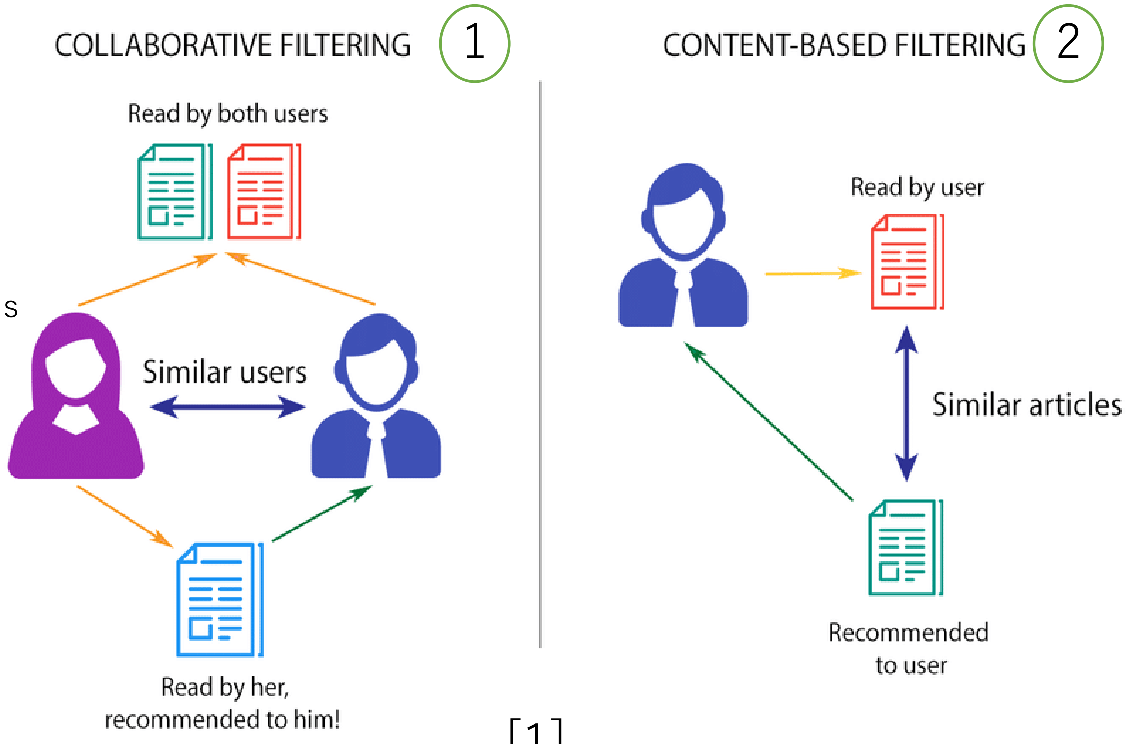
1. Part1: Input data
2. Part2: RNN model
3. Part3: Training
4. Part4: Item sampling
5. Part5: Recommending steps

Background

Recommendation Algorithms:



(item-based)

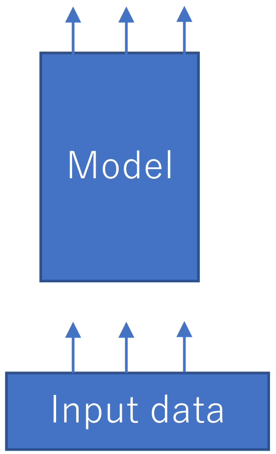


[1]

(user-based)

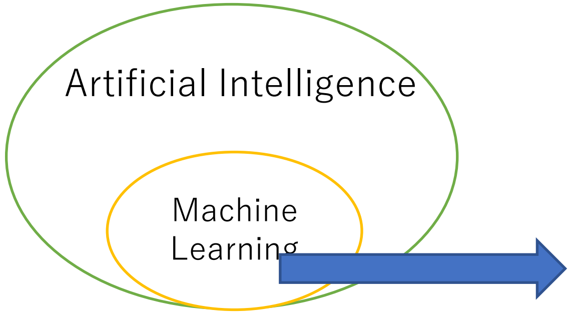
Deep Learning Models

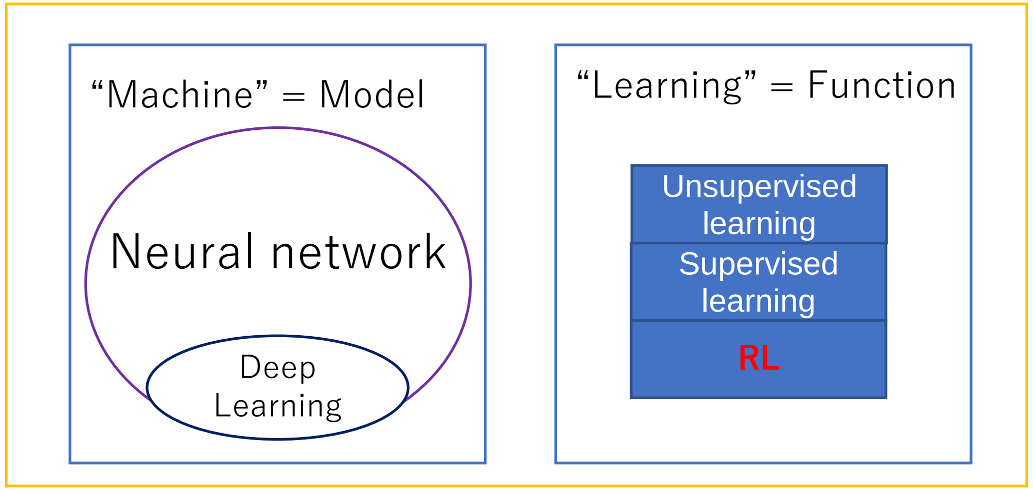
Predict: click or not



[1]TONDJI, LIONEL NGOUPEYOU. "Web recommender system for job seeking and recruiting." (2018).

Reinforcement Learning(RL)





[2]

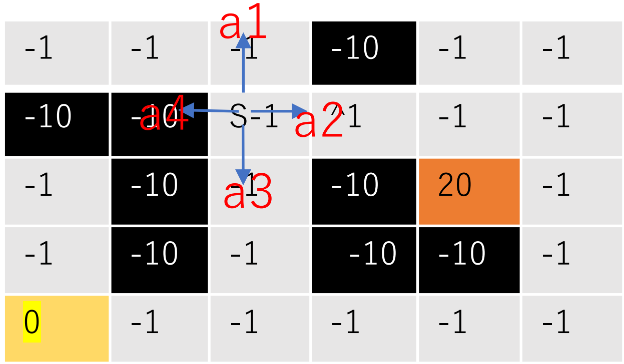
S1: state,
a1,a2,a3,a4: actions

Two major RL types:

valued-based、policy-based

Q-learning: update **Q value** table

action / state	a1	a2	a3	a4
S1	Q(S1, a1)	Q(S1, a2)	Q(S1, a3)	Q(S1, a4)



$$Q(S,A) \leftarrow (1-\alpha)Q(S,A) + \alpha[R(S,a) + \gamma \max_{a} Q(S',a)]$$

Policy Gradient: update **policy** by gradient descent

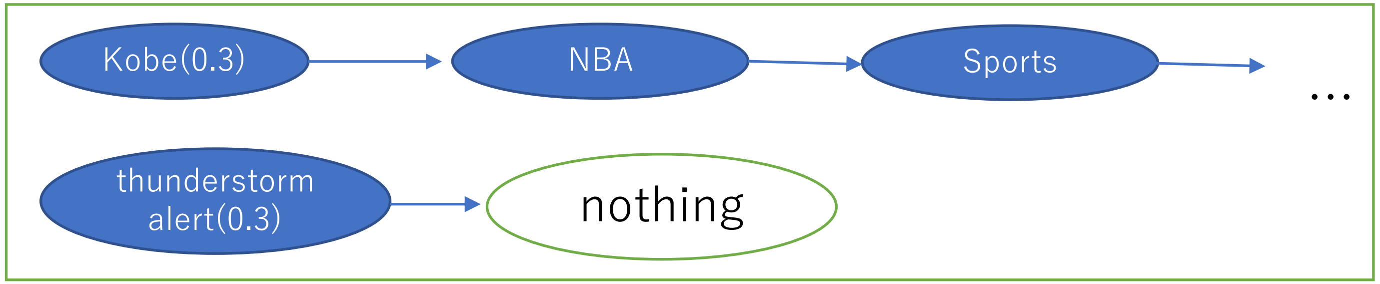
$$\mathrm{E}_{\tau\sim\pi_\theta}[R(\tau)\nabla_\theta \log\pi_\theta(\tau)]$$

[2]Kubo, Takahiro. *Paison De Manabu Kyoka Gakushu: Nyumon Kara Jissen Made*. Kodansha., 2019.

Reinforcement learning for recommendation system

Reasons:

1. Long term rewards



2. Having some randomness

Probability: [0.1, 0.2, 0.3, 0.4], not always the 4th item be chosen

Example:

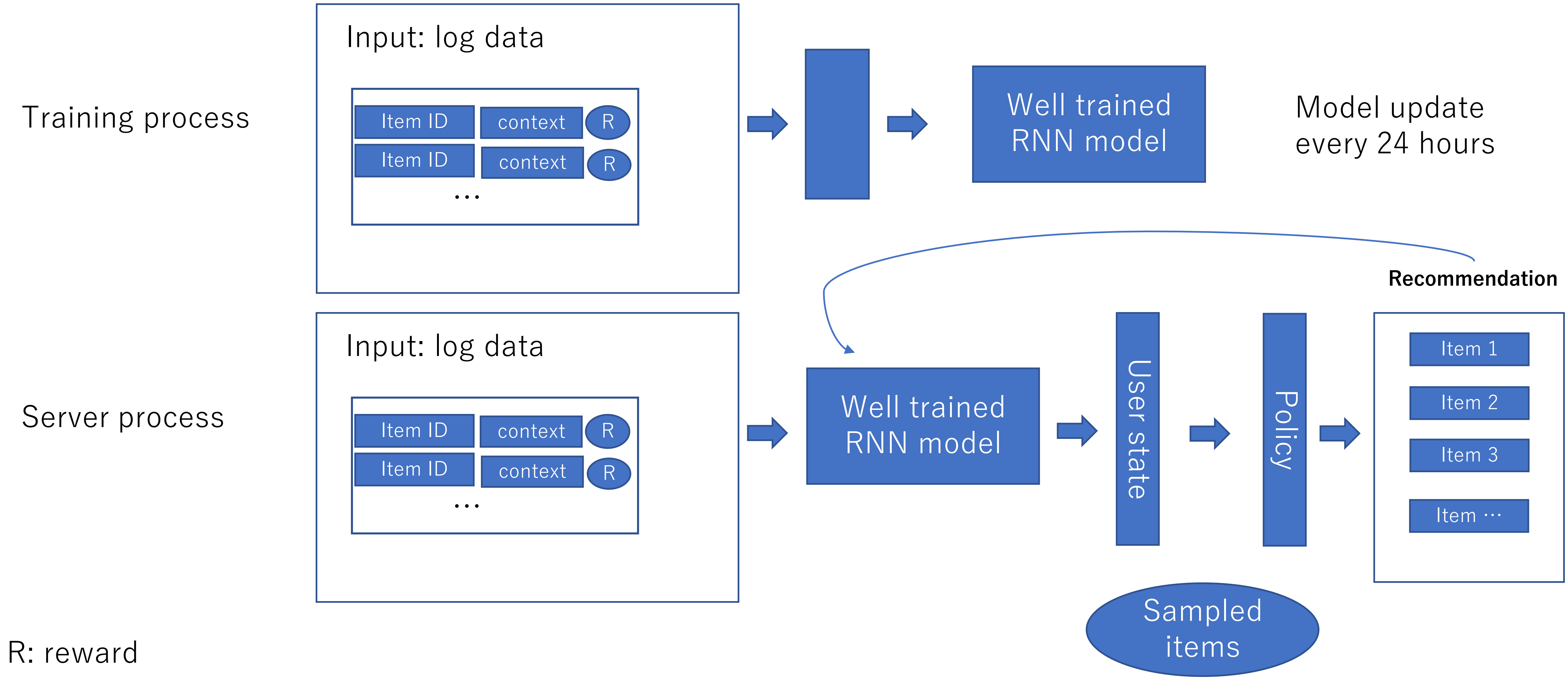
1. <u>Policy Gradient based framework: being used to recommend videos.</u> [3]
2. DQN based framework: being used to recommend news.[4]
3. Critic-Actor based framework: being used to create a virtual environment like virtual Taobao.

1. off-policy
2. Continuous user state
3. Experiment in live experiments

[3]Chen, Minmin, et al. "Top-k off-policy correction for a REINFORCE recommender system." *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 2019.

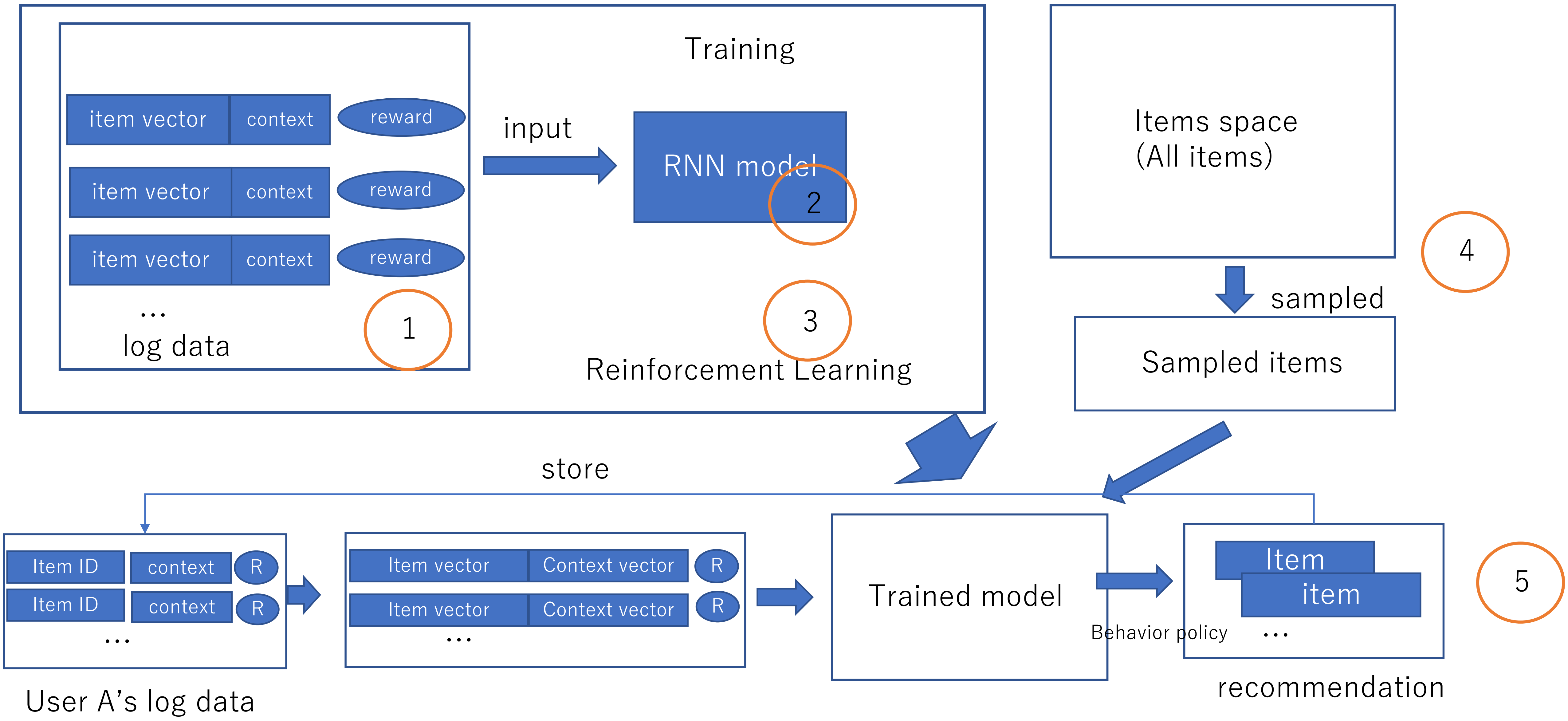
[4]Zheng, Guanjie, et al. "DRN: A deep reinforcement learning framework for news recommendation." *Proceedings of the 2018 World Wide Web Conference*. 2018.

Policy Gradient based Recommendation System

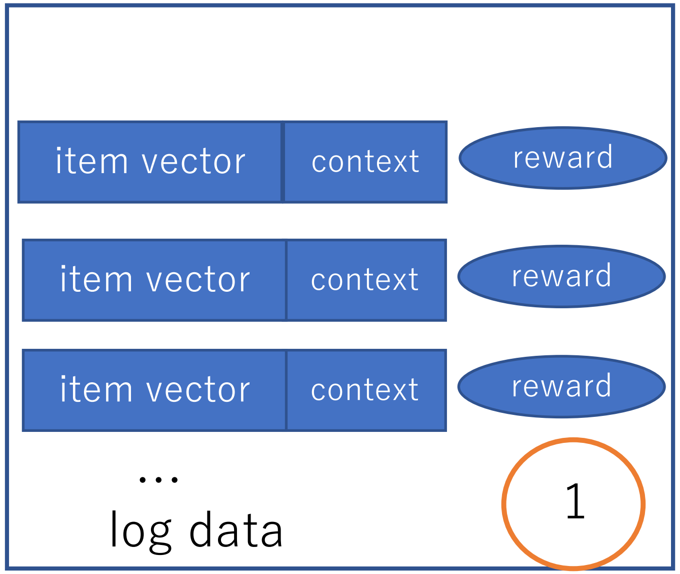


System Structure

System Structure



Part1: Input data



- Item vector:

  Example：カジュアルコンフォート。【春夏生地】メリノウールにポリエステルを混紡した丈夫でしわになりにくい素材です。 48000。

  Embedding: Word2vec/Bert

- Context data:

  Example：timing、device

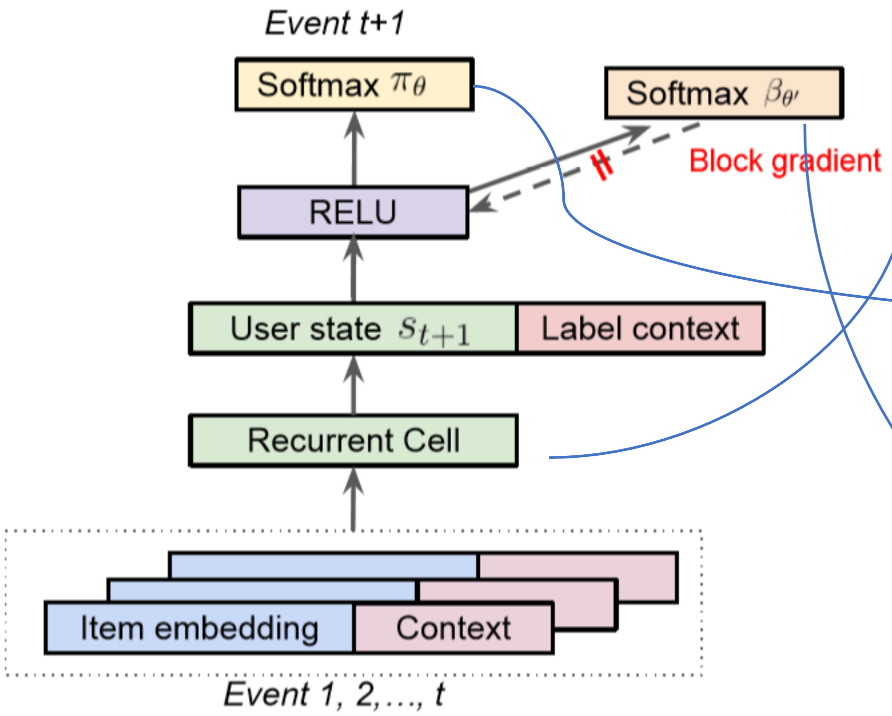
- Reward:

  Example：1.click: 5 point, 2.buy: 15 point
  3.non-feedback: 0 point

Part2: Using RNN model to get user state and policy

RNN model 2

s: state
A: whole item space
a: one item
u_a: item embedding + context vector
T: temperature(0~1)
v_a : item embedding



[3]

CFN cell

$$s_{t+1} = z_t \odot \tanh(s_t) + i_t \odot \tanh(W_a u_{a_t})$$
$$z_t = \sigma(U_z s_t + W_z u_{a_t} + b_z)$$
$$i_t = \sigma(U_i s_t + W_i u_{a_t} + b_i) \qquad [2]$$

π_θ

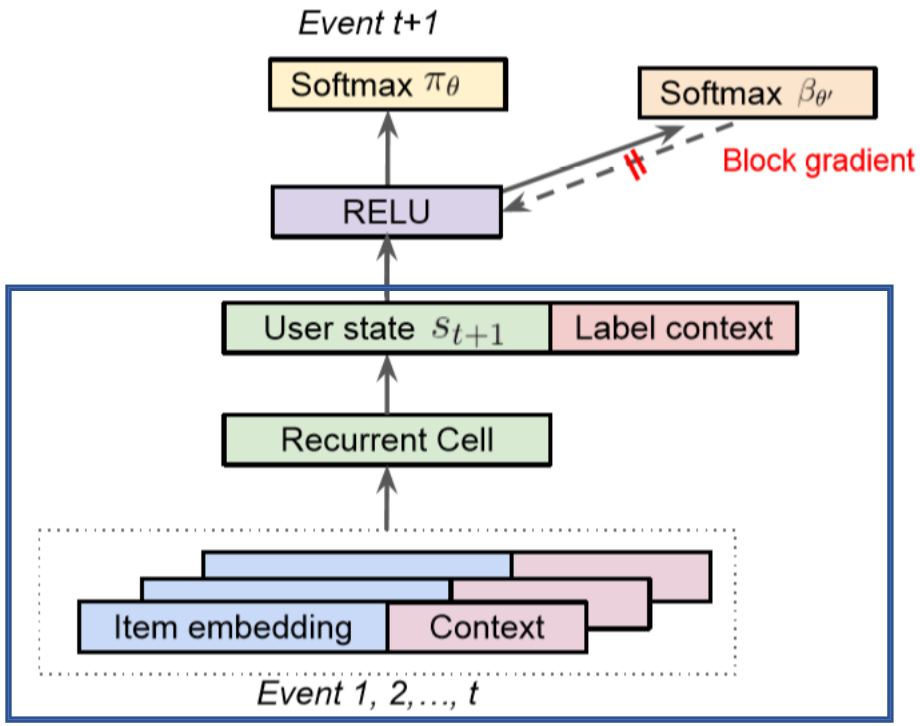
$$\pi_\theta(a|s) = \frac{\exp(s^T v_a/T)}{\sum_{a'\in A}\exp(s^T v_{a'}/T)} \qquad [2]$$

$\beta_{\theta'}$ (*behavior policy*)

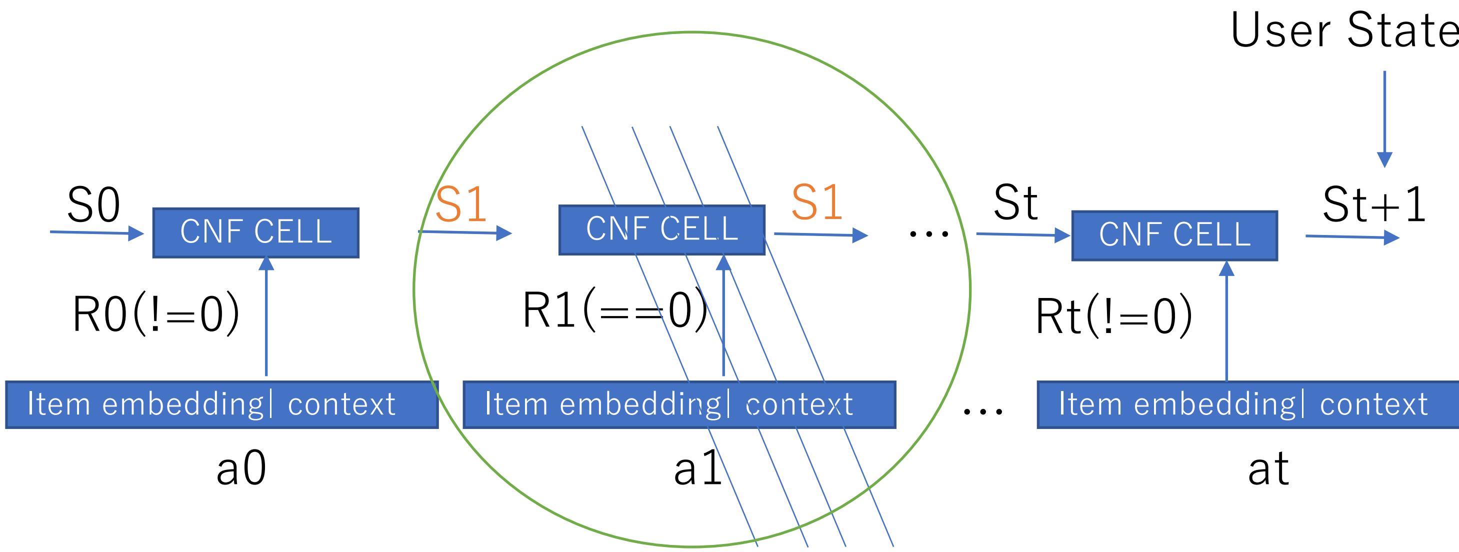
$$\beta_{\theta'}(A|s) = \frac{\exp(s^T v_A/T)}{\sum_{a'\in A}\exp(s^T v_{a'}/T)}$$

Part2: Ignoring non-reward item

RNN model 2



[3]

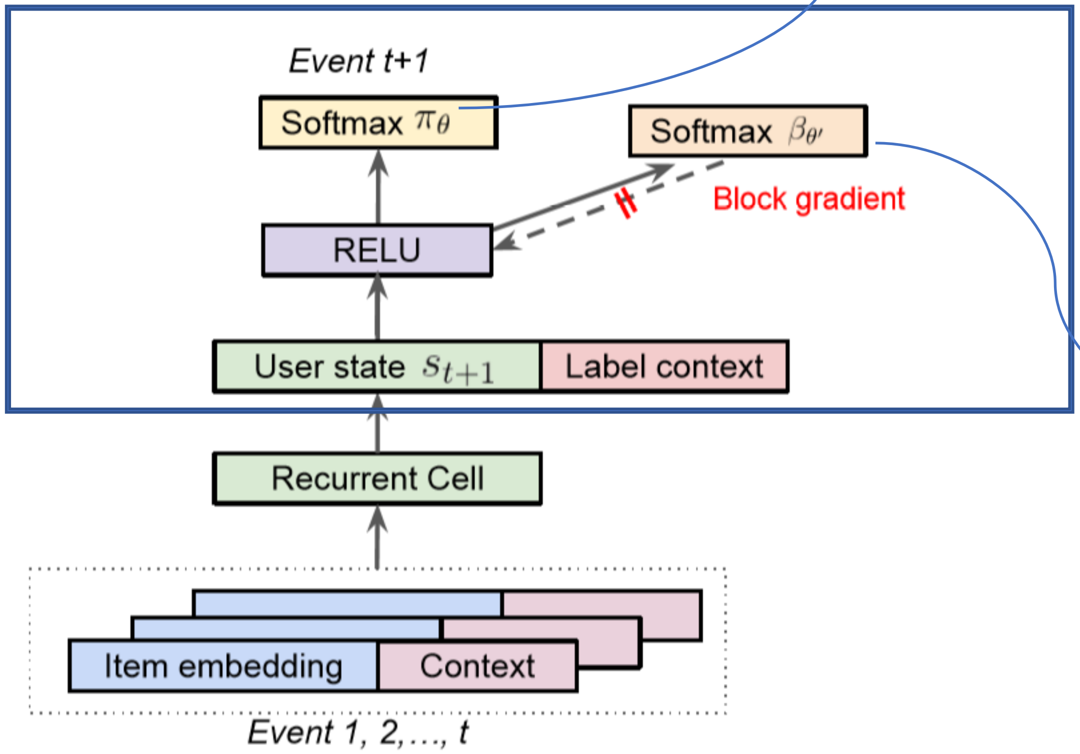


Ignoring non-reward item

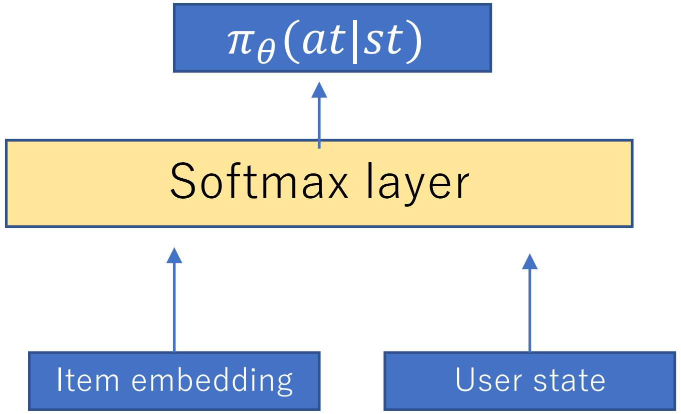
*S0 : [0,0,0,⋯,0]

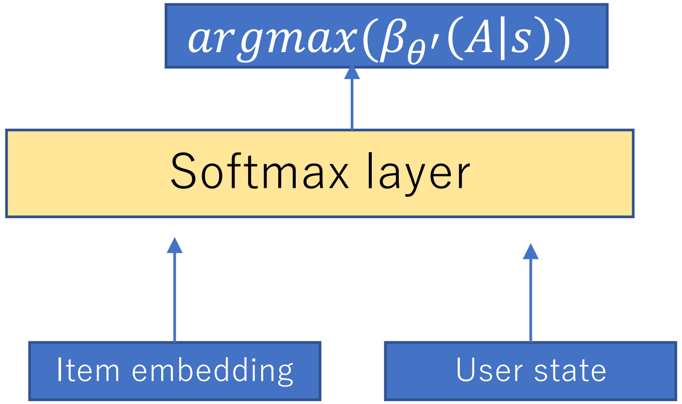
Part2: Computing π_θ

RNN model 2



[3]





教師ありでトレニーング

Part3: Training

Reinforce algorithm:

Trajectory: (s0,a0,s1,a1,..,sn,an)

Gradient

Policy

$$\mathrm{E}_{\tau\sim\pi_\theta}[R(\tau)\nabla_\theta log\pi_\theta(\tau)]$$

Reward

Off policy:

Important weight of the off-policy-corrected gradient estimator

$$\sum_{\tau\sim\beta}[\sum_{t=0}^{|\tau|}\frac{\pi_\theta(a_t|s_t)}{\beta\,(a_t|s_t)}R_t\nabla_\theta log\pi_\theta(a_t|s_t)]$$

Top K:

$$\sum_{\tau\sim\beta}[\sum_{t=0}^{|\tau|}\frac{\alpha_\theta(a_t|s_t)}{\beta\ (a_t|s_t)}R_t\nabla_\theta log\alpha_\theta(a_t|s_t)]$$

$$\lambda_{K(s_t,\,a_t)}=\frac{\partial_\alpha(a_t|s_t)}{\partial_\pi(a_t|s_t)}=K(1-\pi_\theta(a_t|s_t))^{K-1}$$

$$=\sum_{\tau\sim\beta}[\sum_{t=0}^{|\tau|}\frac{\pi_\theta(a_t|s_t)}{\beta\ (a_t|s_t)}\frac{\partial_\alpha(a_t|s_t)}{\partial_\pi(a_t|s_t)}R_t\nabla_\theta log\pi_\theta(a_t|s_t)]$$

Final training expression:

$$\sum_{\tau\sim\beta}[\sum_{t=0}^{|\tau|}\frac{\pi_\theta(a_t|s_t)}{\beta\ (a_t|s_t)}K(1-\pi_\theta(a_t|s_t))^{K-1}R_t\nabla_\theta log\pi_\theta(a_t|s_t)]$$

Arithmer

Part4: data sampling

Items space
(All items)

sampled

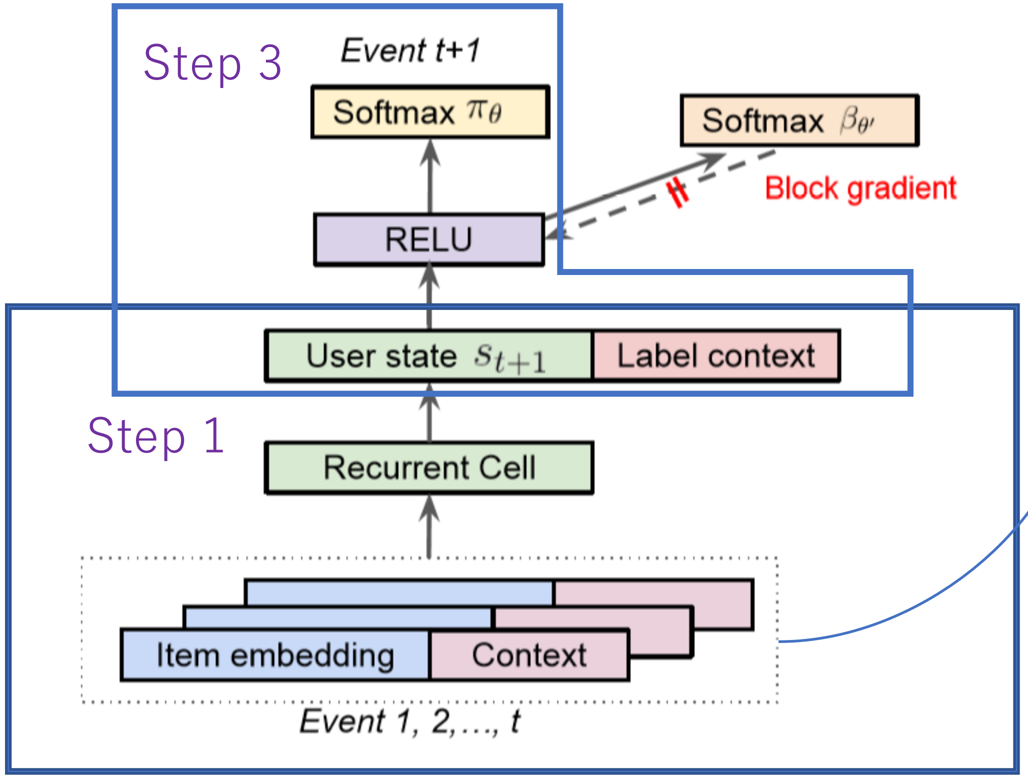
Sampled items

4

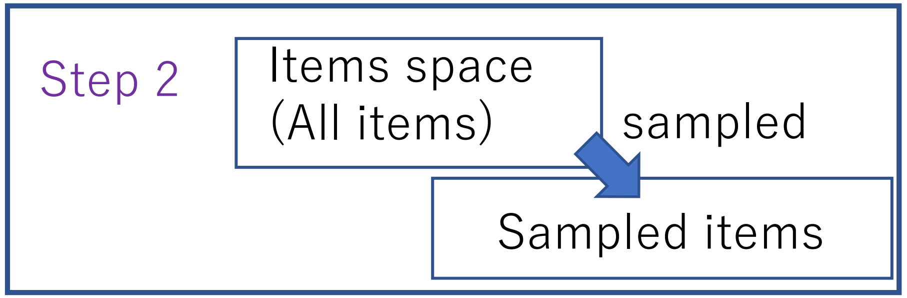
During server time:

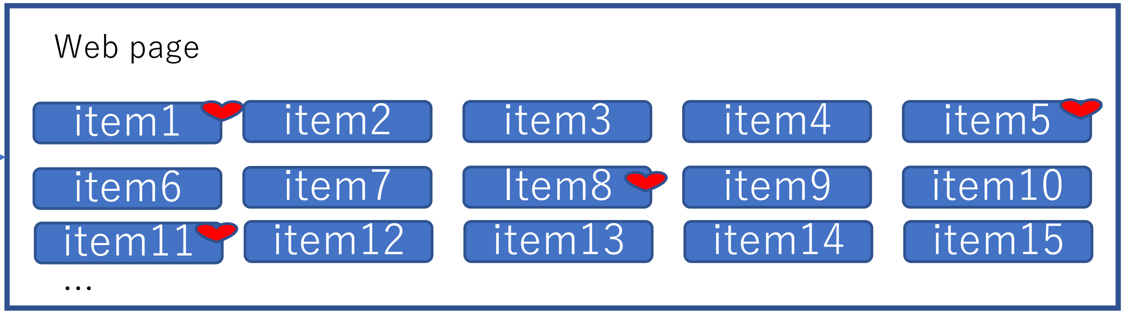
Efficient approximate nearest neighbor-based systems

Part5: Recommendation(1st time)



[3]





*30 popular items from each category

Step1: Choosing 10 items and then get user's state vector.

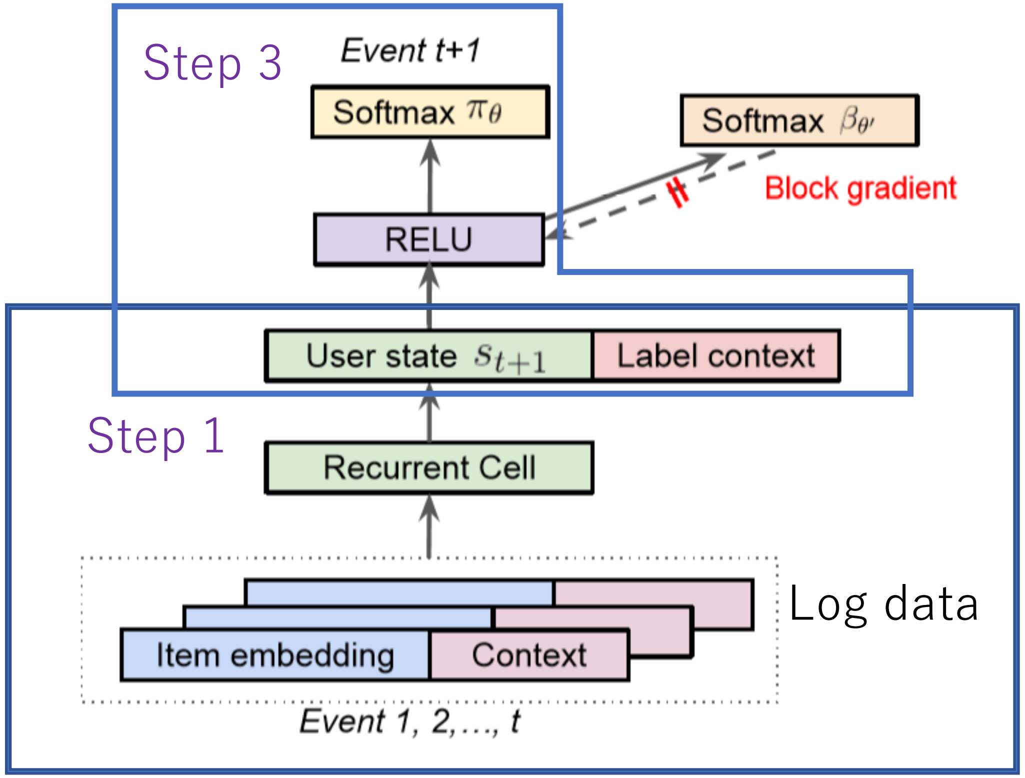
Step2: Sampling items from items space.

Step3: Calculating recommendation probability of all sampled items.

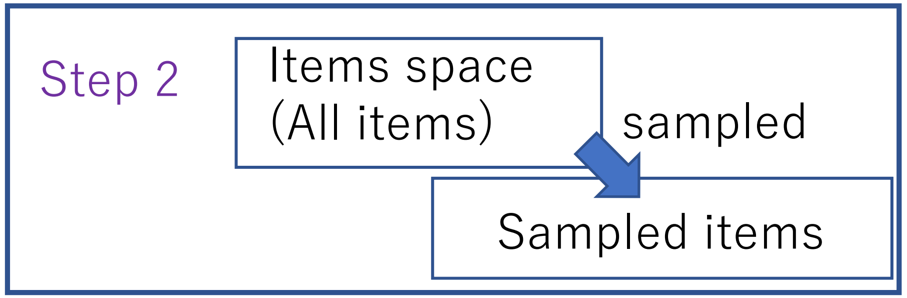
Step4: Randomly recommend K items with recommendation probability.

Step5: Storing recommended item info , context info and users' feedback.

Part5: Recommendation



[3]



Step1: Getting user's state vector by inputting log data.

Step2: Sampling items from items space.

Step3: Calculating recommendation probability of all sampled items.

Step4: Randomly recommend K items with recommendation probability.

Step5: Storing recommended item info , context info and users' feedback.

Arithmer 株式会社
〒106-6040
東京都港区六本木一丁目６番１号 泉ガーデンタワー 38/40F(受付)
03-5579-6683
https://arithmer.co.jp/

Arithmer