Reinforcement Learning in Recommender Systems

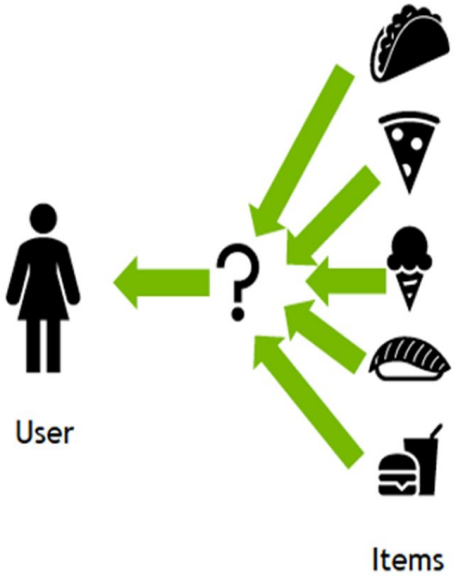
Team

Chichirau Vlad
Ghadamiyan Lida
Ionescu Diana
Oanea Smit
Ouatu Bogdan
Popa Larisa

Introduction

Reinforcement learning in Recommender Systems :

- **interactive recommendation**
- conversational recommendation
- **sequential recommendation**
- explainable recommendation

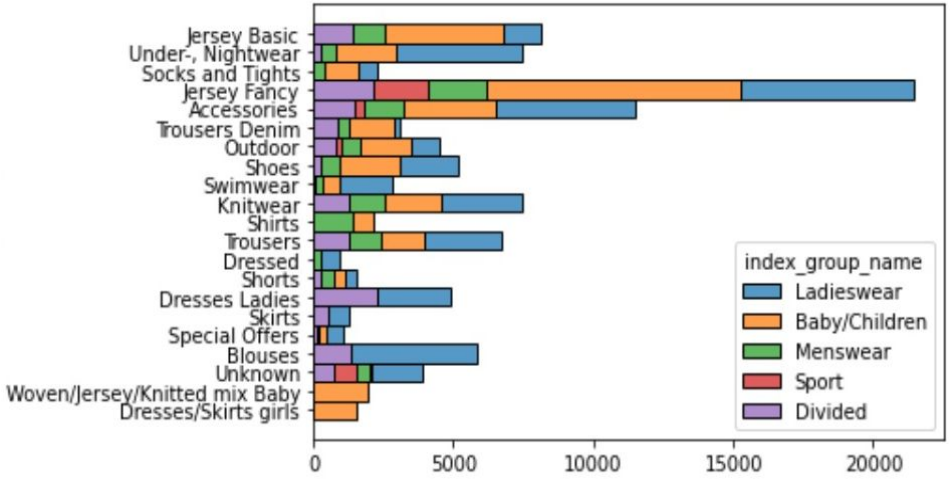


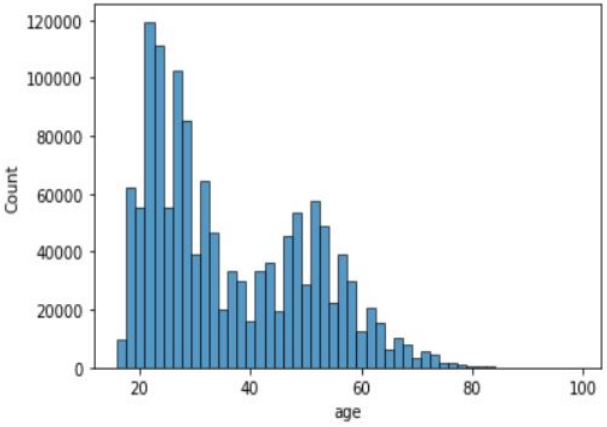
Purpose:

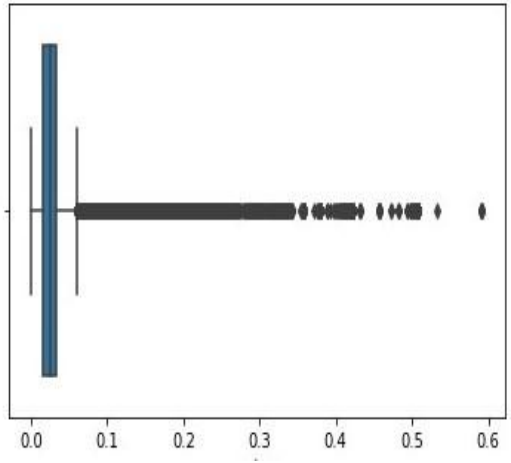
- Study the problem of recommendation system in offline reinforcement learning
- Provide empirical evaluation in the contextual bandit framework

Dataset

H&M Personalized Fashion Recommendation







Try at Home Data Set

This data set contains historical data regarding the purchases and returns from the“try at home” business model which consists in providing the users with garments that they can buy or return - (user_id, product_id, date_received, returned?)

The dataset also contains product features such as color, fabric, shape.

Base Line - Popularity Based Filtering

For the baseline solution we used Popularity Based Filtering which predicts the products that are in trend at the moment based on the purchases history of the last three months.

Therefore, we recommended the first 12 most popular items to every user, achieving the baseline of 0.00304 MAP@12 on the competition leaderboard.

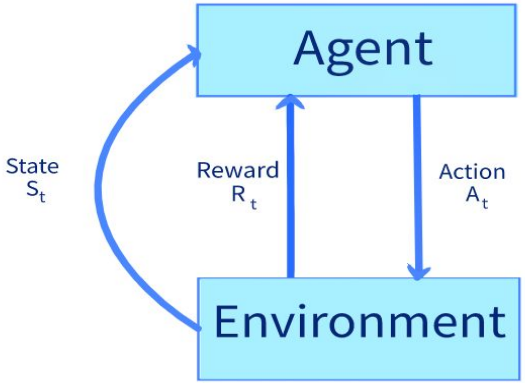
The same strategy results in an 0.01602 MAP@12 on the Try at Home dataset

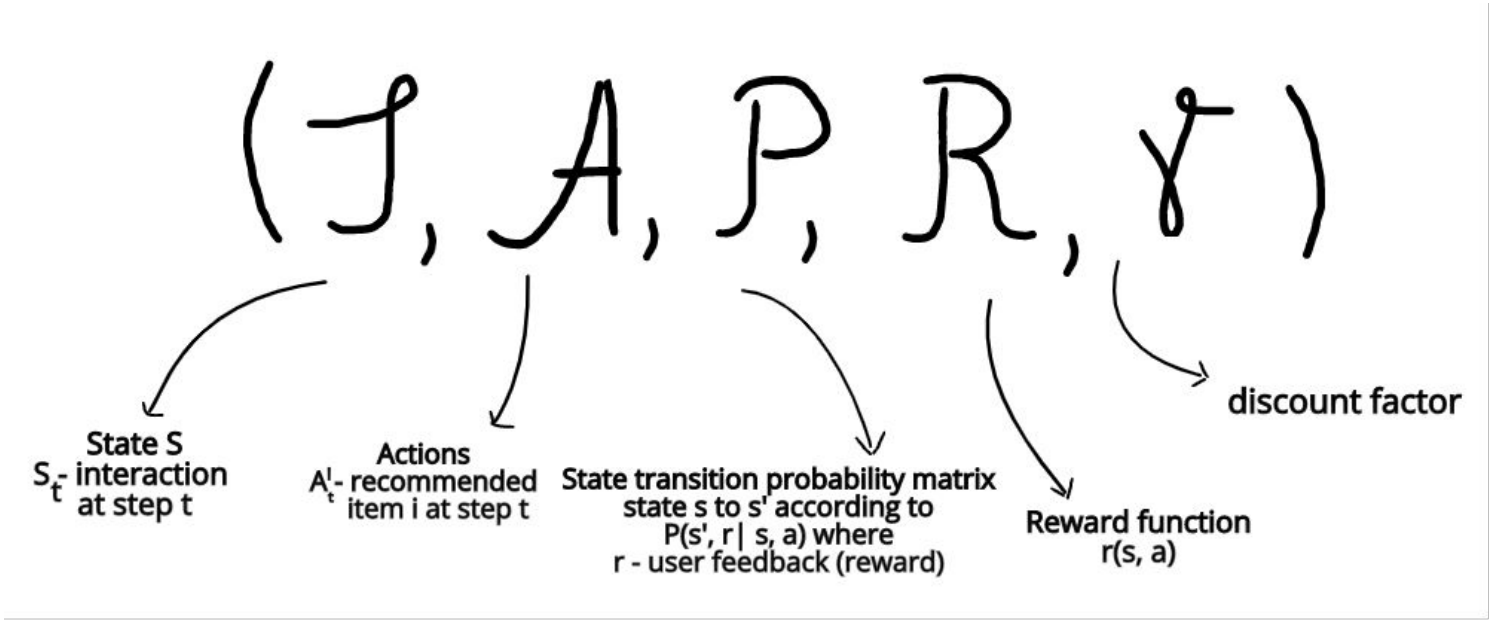
Recommender system context

Modelling: Markov Decision Process

Goal: learn an optimum policy in order to maximize cumulative benefit

- The recommender agent interacts with the environment at each step by recommending an article in the user's current state.
- The environment provides input to the recommender agent in a new state, and proposes a new item to the consumer.




(S, A, P, R, γ)
State S
S_t - interaction at step t
Actions
A^i_t - recommended item i at step t
State transition probability matrix
state s to s' according to
P(s', r| s, a) where
r - user feedback (reward)
Reward function
r(s, a)
discount factor

Actor-Critic based high performing model analysis

We analyzed and compared two recent high performing models.

Supervised Advantage Actor-Critic for Recommender Systems

Xin Xin
School of Computer Science, Shandong University, China
xinxin@sdu.edu.cn

Alexandros Karatzoglou
Google Research, London, UK
alexkz@google.com

Ioannis Arapakis
Telefonica Research, Barcelona, Spain
ioannis.arapakis@telefonica.com

Joemon M. Jose
School of Computing Science, University of Glasgow, UK
Joemon.Jose@glasgow.ac.uk

Self-Supervised Reinforcement Learning for Recommender Systems

Xin Xin*
University of Glasgow
x.xin.1@research.gla.ac.uk

Alexandros Karatzoglou
Google Research, London
alexkz@google.com

Ioannis Arapakis
Telefonica Research, Barcelona
arapakis.ioannis@gmail.com

Joemon M. Jose
University of Glasgow
Joemon.Jose@glasgow.ac.uk

Actor-Critic based high performing model analysis

-Recent papers: 11 Jun 2020 and 5 November 2021

-Both papers present already-existent approaches in the field of RS (RNNs, CNNs, RL approaches)

-New approach for RS: self-supervised reinforcement learning for sequential recommendation tasks

-The first paper: Self-Supervised Qlearning (SQN) and Self-Supervised Actor-Critic (SAC).

-The second paper: Supervised Negative Q-learning (SNQN) and Supervised Advantage Actor-Critic (SA2C).

Actor-Critic based high performing model analysis

-both models 4 state-of-the-art sequential recommendation architectures (GRU, Caser, NItNet and SASRec)

-the main difference between the two papers is that the second one uses negative sampling:

Algorithm 1 Training procedure of SQN

Input: user-item interaction sequence set X, recommendation model G, reinforcement head Q, supervised head

Output: all parameters in the learning space Θ

Initialize all trainable parameters

Create G' and Q' as copies of G and Q, respectively

repeat

Draw a mini-batch of $(x_{1:t}, a_t)$ from X, set rewards r

$s_t = \mathrm{G}(x_{1:t})$, $s_t = \mathrm{G'}(x_{1:t})$

$s_{t+1} = \mathrm{G}(x_{1:t+1})$, $s'_{t+1} = \mathrm{G'}(x_{1:t+1})$

Algorithm 2 Training procedure of SNQN

Input: user-item interaction sequence set X, recommendation model G(·), reinforcement head Q(·), supervised head f(·), pre-defined reward function r(s,a)

Output: all parameters in the learning space Θ

Initialize all trainable parameters

Create G'(·) and Q'(·) as copies of G(·) and Q(·), respectively

repeat

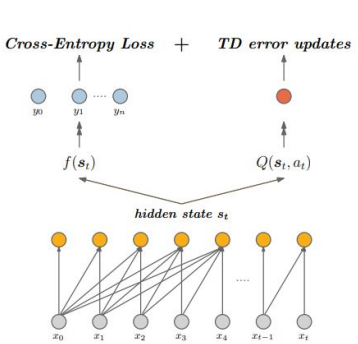
Draw a mini-batch of $(x_{1:t}, a_t^+)$ from X

Draw negative actions set N_t for $x_{1:t}$

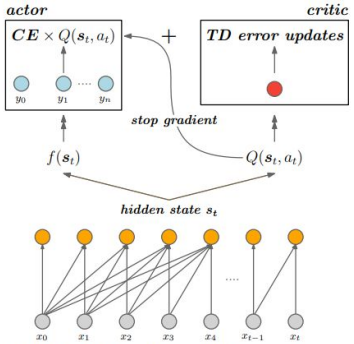
$s_t = \mathrm{G}(x_{1:t})$, $s_t = \mathrm{G'}(x_{1:t})$

$s_{t+1} = \mathrm{G}(x_{1:t+1})$, $s'_{t+1} = \mathrm{G'}(x_{1:t+1})$

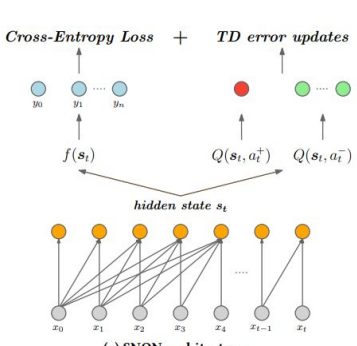
Actor-Critic based high performing model analysis



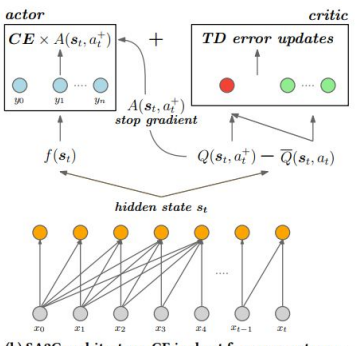
(b) SQN architecture.



(c) SAC architecture. CE stands for cross-entropy.



(a) SNQN architecture.



(b) SA2C architecture. CE is short for cross-entropy.

Actor-Critic based high performing model analysis

-Results: they use to metrics to evaluate the results: Hit Ratio when it comes to clicks done by the user on a specific item (HR) and Normalized Discounted Cumulative Gain (NDCG)

-the most recent paper outperforms the previous one and the simple non-RL architectures

Actor-Critic - Experiment - 1

- Creating the Environment
  - Embed the customers and items data (30-dimensional)
  - Use the transaction history as the backbone of the environment
    - We use a *current_pos* which represent the index in transaction history
    - On reset(): we set *current_pos* = 0 and return *user_embed*
    - On step(action): we compute Cosine Similarity between item purchased in translation history at *current_pos* and the action(12 recommended items) as *reward* - increment *current_pos,* return new *user_embed and reward*

Actor-Critic - Experiment - 2

Actor and Critic are simple NNs

Training:

-reset the environment

-start stepping into the environment and save each state and action in a *replay* buffer

-sample from the *replay* buffer a batch and train the Actor and Critic

Training for a fraction of the transaction history(0.1%) we get a final score of 0.00064 on the competition.

Policy Search -KERL

Purpose:

➔ create a reinforcement learning model that makes use of KG, given the user history and KG

MDP Process:

learn a stochastic policy that maximizes the expected cumulative reward for all uses

State:

-information regarding the users history interactions and information from KG(users, items, interaction)
-state representations:

Sequence-level

- create sequential characteristics of user preference
- RNN for encoding the history sequence of interactions

Knowledge-level

- exploitation: current preference : average items embeddings
- exploration: future preference : Linear Regression

Action:

- select an item at time t+1 for recommendation
- probability: Softmax over item embeddings and state embeddings

Policy:

- input: state
- output: distribution over all possible actions

Reward:

- sequence-level: BLEU score
- Knowledge-level: cosine similarity

Discount Factor: 0.9

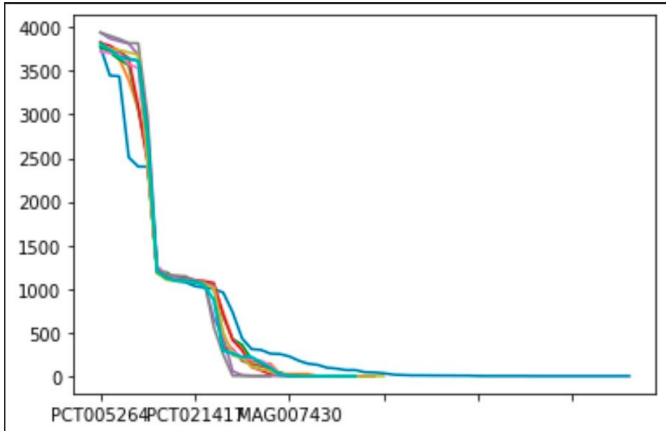
Usage-Based Web Recommendations - UWB

- Interactive recommendation scenario
- Environment:
  - States - the last *w* pages that the user requested
  - Action - a recommendation of an item
  - Reward - based on *w* recommendations made until the current step
- Q-learning problem
  - Q-values – the evaluation of performing an action in a state
- Training
    - the values of each pair *Q(s,a)* is set to zero.
    - the next step is a loop that is repeated until it reaches convergence:
      - choose randomly an episode from the training set
      - set *s* – the first state of the episode
      - for each state of the episode:
        - choose action *a* using ε-greedy policy
        - apply action a, observe next state and compute *r(s,a)*
        - update *Q(s,a)*
        - $s \leftarrow s'$

Bandit context

The problem:

- Classical supervised methods do not correct for selection biases of old policies
- Get stuck in local optima
- Preemptively learn to recommend items with a very short tail distribution



Bandit context

We analyzed mainly three papers

- Offline policy evaluation in a contextual bandit problem - Sang Hoon Kim
- Learning from Logged Implicit Exploration Data - J. Langford
- Exploration Scavenging - J. Langford

Offline policy evaluation in a contextual bandit problem

Purpose:

- Have an offline policy evaluator

Contextual bandit Process:

- The world produces some context $x \in X$
- The learner chooses an action $a \in A$
- The world reacts with reward $r_a \in [0, 1]$

Offline policy evaluation in a contextual bandit problem

Three main solutions:

- Direct Method - estimate reward function
- Inverse Propensity Score - estimate D(action|context) of the new policy
- Doubly robust estimator - combines the previous two approaches

Learning from Logged Implicit Exploration Data

Purpose:

- Train a policy on offline data generated by an old, non-specified policy

Main method:

- model the old policy as probabilistic.
- estimate how likely the old policy is to chose some reward
- For each event, we create a synthetic contextual bandit event
- apply an offline contextual bandit algorithm.

Learning from Logged Implicit Exploration Data

Empirical evaluation

Setting:

- items constitute contexts,
- possible actions is to send an item to one of the users
- reward is 1 if the user keeps the product, or 0 otherwise

Method	Result
Send only the most popular	0.016
Collaborative filtering	0.015
Bandit approach	0.011

No. items in top 12 recommended/user	Result
Send only the most popular	1
Collaborative filtering	0.78
Bandit approach	0.62

- Performs worse than the baseline, but it's expected to
- These are proxy results for the new policy performance
- The strategy is combating popularity bias

Conclusion

- Documented and analyzed broadly the literature on reinforcement learning to address the recommender problem
- Generated two empirical approaches
- Implemented a RL recommendation algorithm for the H&M dataset
- Highlighted the offline evaluation bias problem
- Modeled the Try at Home problem as a bandit problem and applied offline evaluation methods