MOVIE RECOMMENDATION USING DEEP REINFORCEMENT LEARNING

18XD86 – REINFORCEMENT LEARNING LAB

PROJECT WORK

BHARATHI A (18PD05)

SHREENIDHI N (18PD33)

MAY 2022

DEPARTMENT OF APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCES

PSG COLLEGE OF TECHNOLOGY
(Autonomous Institution)
COIMBATORE – 641 004.

CONTENTS

ABSTRACT

Recommender Systems are the systems designed to that are designed to recommend things to the user based on many different factors. These systems predict the most likely product that the users are most likely to purchase and are of interest to. Recommendations typically speed up searches and make it easier for users to access content they're interested in, and surprise them with offers they would have never searched for. In this project work, we explore the use of Reinforcement Learning based techniques to solve the problem of Movie Recommendation. We have implemented the following strategies: Multi Armed Bandits based recommender and an Actor-Critic based recommender framework using Deep Reinforcement Learning.

INTRODUCTION

In this project, we implement several reinforcement learning based algorithms to solve the problem of recommending movies to users based on their previous ratings and interests.

1.1 RECOMMENDER SYSTEMS

The purpose of a recommender system is to suggest relevant items to users. Recommender systems are one of the most successful and widespread application of machine learning technologies in business. They are applied in scenarios where users interact with various items.

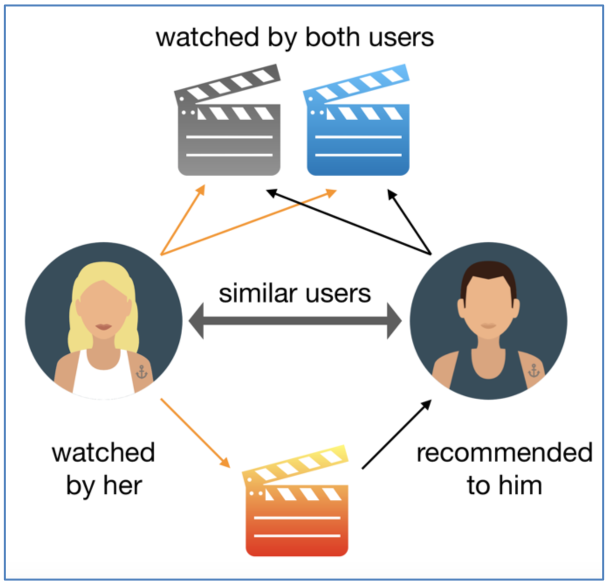

Figure 1.1 Recommendation Idea

Machine learning algorithms in recommender systems are typically classified into two categories — content based and collaborative filtering methods although modern recommenders combine both approaches. Content based methods are based on similarity of item attributes and collaborative methods calculate similarity from interactions.

1.2 NEED FOR REINFORCEMENT LEARNING

The problem with traditional Machine learning based recommender systems is that they consider recommendation as a static process. The model is unable to adapt to the changing preferences of users. Furthermore, the majority of existing recommender systems are designed to maximize the immediate (short-term) reward of recommendations, i.e., to make users order the recommended items, while completely overlooking whether these recommended items will lead to more likely or more profitable (long-term) rewards in the future. Reinforcement based recommender systems Continuously improve strategies during the interactions with users and automatically learn the optimal strategies via recommending trial-and-error items and receiving reinforcements of these items from users' feedbacks.

DATASET

A description of the movielens dataset, which has been used in the project along with the data preprocessing steps are discussed in this chapter.

2.1 DATASET DESCRIPTION

The MovieLens datasets, first released in 1998, describe people's expressed preferences for movies. These preferences take the form of tuples, each the result of a person expressing a preference (a 0–5 star rating) for a movie at a particular time. There have been four MovieLens datasets released, known as 100k, 1m, 10m, and 20m, reflecting the approximate number of ratings in each dataset. They each represent rating tuples of the form <user, item, rating, timestamp>. Movies are listed along with their genres. The 100k and 1m datasets include users' demographic data, while the 10m and 20m datasets do not include any demographic information. This project uses the 1m dataset for bandit experiments and the 100-k dataset for deep learning experiments due to the scarce of computational resources in dealing with large datasets.

2.2 DATA PROCESSING

The userId, MovieId, Ratings, Liked and Timestamp attributes are extracted from both the 100-k and 1-m datasets. A historical representation is built for each user ID by storing their previously rated movies. This creates a context, that is, the list of previously rated movies for each user and this context which contains a sequence of movies is used to train a Neural network model to learn the embeddings for the movies. So, instead of using integer representation, the embedding representation is used for each user.

METHODOLOGY

The various bandit and deep reinforcement learning experiments carried out are detailed in this chapter.

3.1 MDP FORMULATION

The sequential interactions between users and a movie recommender system is modelled as a **Markov Decision Process (MDP)** and Reinforcement Learning is applied to automatically learn the optimal strategy. Given the historical MDP, i.e., (**S, A, P, R, γ**), the goal is to find a recommendation policy **$\pi : S \rightarrow A$,** which can maximize the cumulative reward for the recommender system.

S -> State space where each state represents a list of movies that a user has previously rated

A -> Action space where each action denotes a list of movies recommended to the user

P – Transition probability matrix

R – Rewards denoting whether the user liked the recommendation or not.

Γ – Discount factor

3.2 BANDIT MODELS

The objective of building the bandit models is to recommend movies(arms) and maximize the rewards, which represent whether the user 'liked' the movie or not. It is a binary variable denoting 0 or 1. The following bandit algorithms were implemented:

- **UCB1 – Upper Confidence Bound:** the UCB algorithm changes its exploration-exploitation balance as it gathers more knowledge of the environment.
- **EXP3 – Exponential weight algorithm for exploration and exploitation:** It works by maintaining a list of weights for each of the actions, using these weights to decide randomly which action to take next, and increasing (decreasing) the relevant weights when a payoff is good (bad)
- **EXP4 – Exponential weight algorithm for exploration and exploitation with experts:** This algorithm works similar to Exp3 with the addition of expert advice or probability distribution for different actions and instead of assigning scores to actions, this algorithm assigns scores to the experts.

3.3 ACTOR-CRITIC FRAMEWORK

The second approach uses Deep Reinforcement learning using Actor-Critic framework. The objective here is to recommend a list of movies to the user that maximizes the expected ratings. A rating is a discrete values in the range [1, 5]. The Actor-Critic model, a Temporal Difference (TD) version of Policy Gradient is applied to train the parameters that maximize the expected ratings. The following two components are trained:

- **Actor:** The actor scores each action using a state specific scoring function and generates an action for the particular state

- **Critic:** The critic evaluates the action output by the actor by learning an action value function Q(s_t, a_t).

Then, according Q(st , at), the Actor updates its' parameters in a direction of improving performance to generate proper actions in the following iterations. The parameters of both actor and critic are updated using an estimated target actor and target critic since the true action-value function is not known to calculate the error in function approximation.

EXPERIMENTS AND RESULTS

The results generated by the models are detailed in this section.

COMPARISON OF THE BANDIT ALGORITHMS

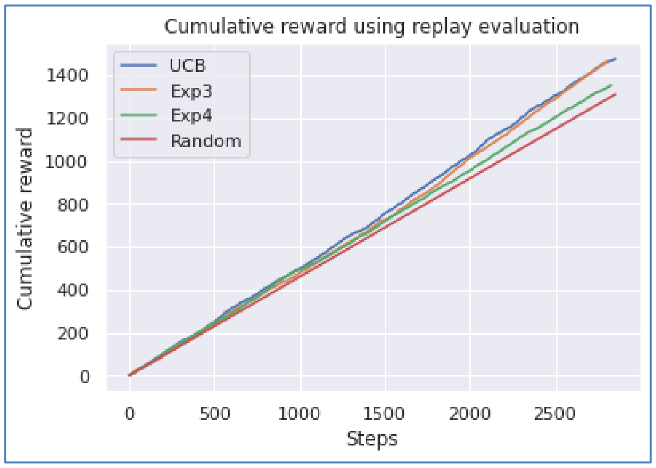

Fig 4.1 Cumulative Rewards for each algorithm

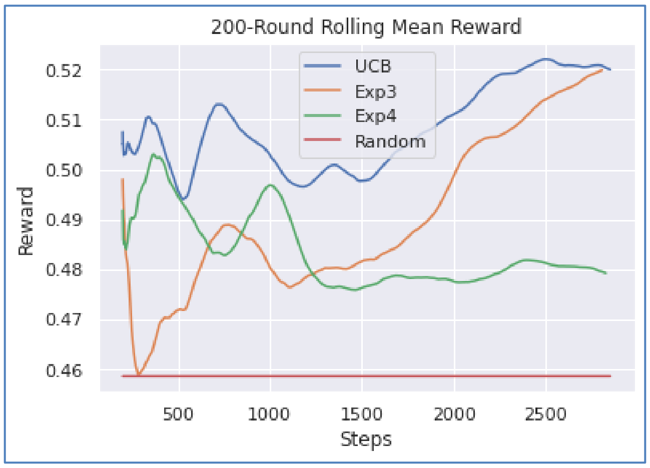

Fig 4.2 Rolling Mean Reward for the algorithms

PREDICTIONS OF THE ACTOR-CRITIC NETWORK

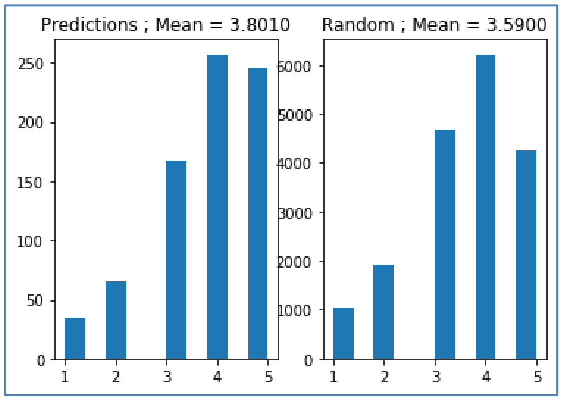

Fig 4.3 Train Predictions by Actor Network

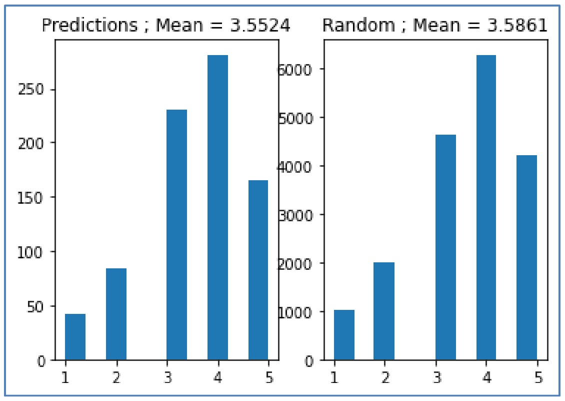

Fig 4.4 Train Predictions by Target Network

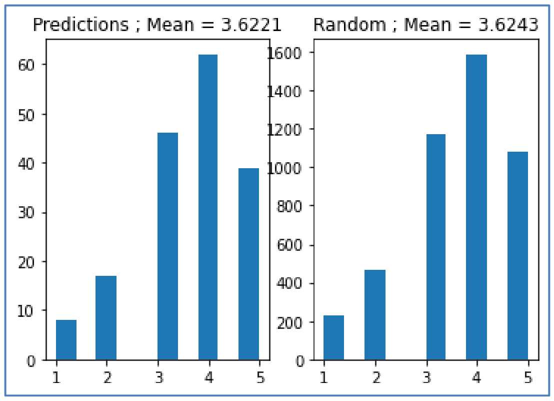

Fig 4.5 Test Predictions by Actor network

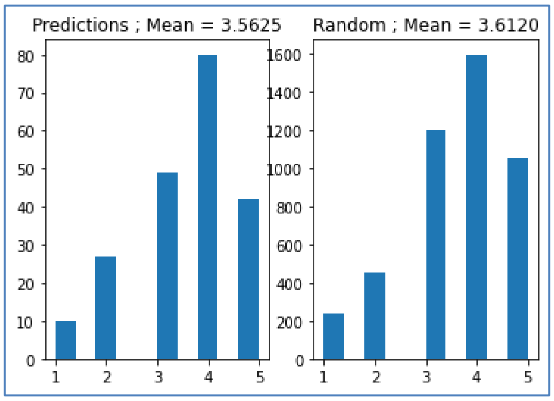

Fig 4.6 Test Predictions by Target Network

The figures 4.3 and 4.4 show the mean ratings predicted by the Actor and Target network on the training set. And, the figures 4.5 and 4.6 show the mean mean ratings predicted by the Actor and Target network on the testing set.

CONCLUSION

From the results of the bandit simulation shown in Fig 4.1 and Fig 4.2, we find that the cumulative rewards given by each of the three algorithms, UCB1, EXP3 and EXP4 are so that UCB1 dominates EXP4 and EXP3 by a small amount. This can partly be attributed to the fact that the number of training steps being less and longer training time. Perhaps, as the training time is further increased, EXP3/4 might emerge outperforming UCB1 since they are theoretically proven better than UCB1. Fig 4.2 shows a similar interpretation, but this time using mean rewards. In the case of Actor-Critic, in both the training and test sets, the mean of the predicted ratings are higher for the Actor network compared to the Target network. The results of Actor-Critic also has much scope for improvement if the training time is increased to several hours.

BIBLIOGRAPHY

[1] Sungwoon Choi, Heonseok Ha, Uiwon Hwang, Chanju Kim, Jung-Woo Ha, and Sungroh Yoon. 2018. Reinforcement Learning based Recommender System using Biclustering Technique. In Proceedings of Workshop on Multidimensional Information Fusion for User Modeling and Personalization (IFUP'18). ACM, New York, NY, USA, 4 pages. https://doi.org/10.475/123_4

[2] Xiangyu Zhao, Liang Zhang, Long Xia, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2019. Deep Reinforcement Learning for List-wise Recommendations. In The 1st Workshop on Deep Reinforcement Learning for Knowledge Discovery (DRL4KDD '19), August 5, 2019, Anchorage, AK, USA. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

[3] Richard S. Sutton, Andrew G. Barto, "Reinforcement Learning: An Introduction"

[4] https://grouplens.org/datasets/movielens/