CSCI 566 - Deep Learning and its Applications (Spring 2024)

Course Project - Pre Proposal Document

Reinforcement Learning for Recommender Systems - Environment Suite

By: Pranav Parnerkar (parnerka@usc.edu)
Assigned TA: Ayush Jain

Background and Idea: Recommender Systems have been a part of the digital ecosystem since the inception and rapid proliferation of search engines. Initially, these systems were easy to implement and interpret as they leveraged mathematically formalized algorithms. Mathematically powered rec-systems were eventually superannuated as they lacked multi-facetedness by design. With the advent of complex deep-learning architectures in the early 2010s, rec-systems pivoted from being interpretable to convoluted black boxes for capturing better dynamics in big data. However, as we prioritize making AI more robust, researchers are looking to incorporate reinforcement learning (often considered the only way to achieve artificial general intelligence) into rec-systems to make them more reliable, interpretable, and potentially better than its hip deep learning-based variants. This change brings a ton of technical challenges with it.

In this project, we aim to tackle one such challenge. Currently, the RL-RecSys community lacks a unified suite of RL-RecSys environments simulating different kinds of RecSys scenarios, like shopping recommendations (Amazon), video recommendations (Netflix, YouTube), educational recommendations (Ed-tech assistants), etc. Our objectives will be to benchmark simple RL agents on a developed suite of environments and identify relevant RL-RecSys assumptions/challenges (in the process).

Why is it useful/interesting: Let's consider a scenario - you have an MLE job interview at OpenAI (congratulations on beating their ATS system, by the way), and as we all do, you are scrambling at the last minute. From mindlessly scrolling Reddit/Medium/LinkedIn posts, you've got to know that interviews in the past have been deep learning heavy. YouTube is your virtual guru, and you start binging on deep learning concepts, tutorials, and mock interviews. After a couple of hours, you realize that your recommendations have saturated, and now you are watching repetitive content (though from different creators). Expressing a momentary dissatisfaction with YouTube, you will subsequently initiate searches for specific content in the search box to avoid wasting time. Eureka - this happens because deep learning-based rec-systems often suffer from two problems - myopia and system-induced bias.
Let's dive into them individually:

1. Myopia: These models try to pigeonhole recommendations that are more likely to lead to an immediate user response (aka click-bait) instead of long-term utility.
2. System-induced bias: These models try to replicate similar click flows based on a population sample's history, thereby inherently inducing bias to one form/flow of content for overall optimization rather than that specific user's preferences.

These problems are difficult to rectify in the current rec-systems because of their black-box nature. Reinforcement learning-based rec-systems, however, would be significantly more interpretable in nature, and if these two problems arise, then they would be easier to pinpoint and address programmatically. An interpretable system also has the potential to tell us how it made a recommendation based on its underlying mathematical foundation, thereby giving us room to explore and build better mechanisms gradually.

Framing the Recommendation Problem into the Reinforcement Learning Setup: The recommendation problem is a unique instance of a reinforcement learning problem whereby the user is the environment upon which the agent, the recommendation system, acts to receive a reward - a click or engagement by the user [3]. Learning is interpretable, wherein a reward/penalty is provided to the recommendation agent for optimizing the underlying model or policy.

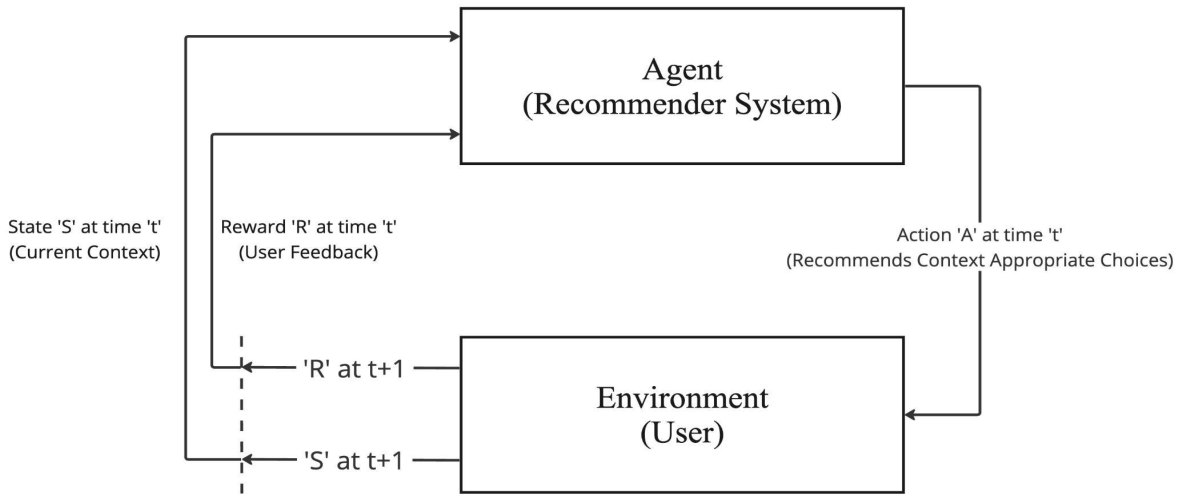

Fig 1: Framing the recommendation problem into the reinforcement learning setup

Proposed Solution: We plan on leveraging RecSim [2] - a configurable recommender systems simulation platform developed by Google Research, to build a unified suite of environments for benchmarking simple RL agents. RecSim follows a plug-and-play paradigm, making it easier to traverse RL-RecSys's steep learning curve and achieve our task of building and simulating environments and agents. It also gives us access to libraries with pre-implemented RL algorithms for validating our environments. One minor shortcoming of RecSim is its inability to ingest custom datasets. By design, it needs to be fed RL-oriented data in a specific way. As a result, we will have to synthesize recommendation data from sources mentioned in the RL4RS [1] paper that fits into the RecSim framework.
Note: *RecSim's GitHub page hosts a reference code for setting up a toy custom environment and agent. It serves as a starting point for us.*

Expected Outcomes:

- Benchmark simple RL agents on a developed suite of environments.
- Identify relevant RL-RecSys assumptions/challenges in the process.

Potential Deliverables till Mid Term Report:

- Set up a toy recommendation environment and agent.
- Familiarize ourselves with components of the RecSim environment.
- Synthesize data that fits into RecSim/look for better alternatives to RecSim.
- Researching benchmarks for existing datasets using preconfigured environments and various RL algorithms.
- Finding robust metrics for our recommendation environment.

References:

[1] Wang, Kai, et al. "Rl4rs: A real-world benchmark for reinforcement learning based recommender system." arXiv preprint arXiv:2110.11073 (2021).
[2] Ie, Eugene, et al. "RecSim: A configurable simulation platform for recommender systems." arXiv preprint arXiv:1909.04847 (2019).
[3] Afsar, Mohammad Mehdi et al. “Reinforcement Learning based Recommender Systems: A Survey.” ACM Computing Surveys 55 (2021): 1 - 38.
[4] Xiaocong Chen et al. “Deep reinforcement learning in recommender systems: A survey and new perspectives”