Notes for: Generative Aversarial User Model for Reinforcement Learning Based Recommendation System

Michael Nefiodovas. *

March 24, 2020

1 Setting

1.1 Intuition

- You work at YouTube and want to recommend YouTube videos to your users to maximise their "utility" from watching videos.
- You have access to a set of all available videos (all YouTube videos) to show the user.
- You must design an algorithm to select a subset of videos to recommend to the user after they watch a video.
- Once shown a selection of recommendations, the user may click on one of the videos to watch or may abstain from clicking.
- You have access to users' historical view data.

Summary: *users are recommended a page of items and they provide feedback, and then the system recommends a new page of items.*

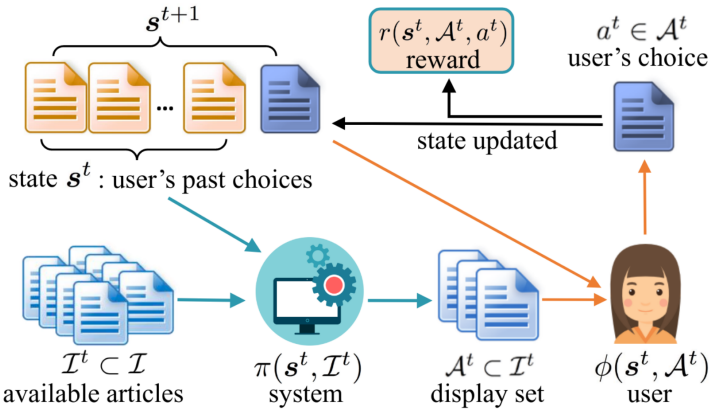
1.2 Assumptions

1. Users are not random and given a set of k items, users will attempt to maximise their own reward/utility r.
2. Watching each video incurs an opportunity cost, if watching a video is a waste of time or mental energy the user can choose to *not* click on any recommendations.

*paper: https://arxiv.org/pdf/1812.10613.pdf

3. Reward is the marginal benefit of taking an action, this means that r depends not only on the video about to be watched but also the user's watch history leading up to this video[1].

2 Framing



Environment

It may be strange to think about it at first but: The "user" in this scenario is the environment which we interact with by selecting videos/documents to recommend. The user obeys their policy $\phi^*(S^t, A^t)$, selecting a choice based on the current state and the recommending agent's selected action (what videos were shown).

State

The current environment state s^t refers to "an ordered sequenceof a user's historical clicks".

The current state can be represented as an embedding of the historical sequence of items clicked by the user.

Action

The action is a selection of videos to present the user, that means if there is $\mathcal{I}$ total available videos to select from and you are showing the user k ideos at a time, then action A^t is selected $A^t \in \binom{\mathcal{I}^{\sqcup}}{k}$. As you can imagine, the possible action space is absolutely huge even for medium sized I^t and k[2].

Reward

$r(s^t, A^t, a^t)$ measures a user's utility or satisfaction after making a choice. A recommendation system should attempt to maximise this. (in theory this reward could also consider company goals to align the recommneder with the company moreso).

[1] Someone might not be interested in Taylor Swift at the beginning but do become interested after listening to it. Users may also get bored if they watch the same video 100 times

[2] This problem is addressed through their *Cascading Q-Network* architecture.

Importantly, measuring a users utility is a hard task[3] as we do not know ϕ^* and often times "synthetic" reward signals are crafted (e.g. +1 reward whenever you click an item).

3 Contribution

3.1 Generative Adversarial User Model

We need a way to estimate ϕ^* so that we can accurately reccomend things to the user.

We have a generator ϕ which tries to mimic the action sequences provided by a real user (note: all users attempt to maximise their reward function r).

We have a discriminator r which tries to differentiate the actual user's actions from the generated actions.

As mentioned before the state can be represented as an embedding of past states. They provide two suggestions on how to to this: via LSTM or via a position weight matrix

They mention somgthing about how the process "may be unstable due to the non-complexity nature of the problem". I didn't really have time to comprehend what these things mean but they say they use a some sort of "special regularization" for initializing the training process.

3.2 Cascading RL Policy for Recommendation

One of the problems was the combinatorial action space $\binom{\mathcal{I}^t}{k}$ They solve this by creating k-many Q functions and applying them in a cascading fashion.

Recomender actions as $A = a_1 : k \subset \mathcal{I}$

Optimal action as $A^* = a^*_{1:k} = \arg\max_A Q^*(s, A)$

$$\max_{a_1:k} Q^*(s, a_{1:k}) = \max_{a_1}(\max_{a_2:k} Q^*(s, a_{1:k}))$$

Each Q-function considers the previous selection to decide the next document/video to select/add to the set which will be recommended to the user.

[3] This problem is addressed through their *Generative Adversarial User Model*