Adapting to Dynamic User Preferences in Recommendation Systems via Deep Reinforcement Learning

By

Petru-Luca Pantea[1]

Responsible Professor:
Dr. Frans A. Oliehoek[1]

Supervisors:
Dr. Aleksander Czechowski[1]
Davide Mambelli[1]
Oussama Azizi[1]

A Dissertation Submitted to EEMCS faculty Delft University of Technology[1],
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 19^{th}, 2022

Adapting to Dynamic User Preferences in Recommendation Systems via Deep Reinforcement Learning

Petru-Luca Pantea[1]
Delft University of Technology, The Netherlands
Department of Intelligent Systems[1]

ABSTRACT

Recommender Systems play a significant part in filtering and efficiently prioritizing relevant information to alleviate the information overload problem and maximize user engagement. Traditional recommender systems employ a static approach towards learning the user's preferences, relying on logged previous interactions with the system, disregarding the sequential nature of the recommendation task and consequently, the user preference shifts occurring across interactions. In this study, we formulate the recommendation task as a *slate* Markov Decision Process (*slate-MDP*) and leverage deep reinforcement learning (DRL) to learn recommendation policies through sequential interactions and maximize user engagement over extended horizons in non-stationary environments. We construct the simulated environment with various degrees of preferential dynamics and benchmark two DRL-based algorithms: FullSlateQ, a non-decomposed full slate Q-learning based on a DQN agent, and SlateQ, which implements DQN using slate decomposition. Our findings suggest that SlateQ outperforms by 10.57% FullSlateQ in non-stationary environments, and that with a moderate discount factor, the algorithms behave myopically and fail to make an appropriate tradeoff to maximize long-term user engagement.

KEYWORDS

Recommender Systems, Reinforcement Learning, User Modelling

1 INTRODUCTION

The past decades have seen an exponential increase in the volume of digital information available and the number of users capable of accessing those resources have created a significant challenge in filtering and efficiently prioritizing relevant information. Recommender systems aim to tackle the information overload problem by effectively modelling the users' preferences, relying on past user behaviour and feedback received via interactions with the system [Aggarwal 2016; Isinkaye et al. 2015; Lu et al. 2012]. User modelling sits at the core of many recommendation systems, as it allows the algorithm to efficiently suggest information that matches the users' preferences with as little latency as possible [Jawaheer et al. 2014].

However, approaches constructed on top of classic recommender, such as Collaborative Filtering (CF) with Implicit Feedback, illustrated by [Hu et al. 2008], or hybrid recommender systems (CF combined with content-based methods), such as HyPER introduced by [Kouki et al. 2015], employ a static approach, and disregard the dynamic, sequence-based decision-making process inherent to real-world recommendation scenarios. [Adomavicius and Zhang 2012] presents how such systems fail to adapt to variations in user preferences based on recent interactions, which ultimately leads to poor long-term user engagement.

To fuse the capability of sequential processing, and to integrate the optimization of user engagement and satisfaction over extended horizons, Deep Reinforcement Learning (RL) is an ideal proposal [Huang et al. 2021; Munemasa et al. 2018; Zhao et al. 2018]. The process of recommendation can be modelled as a *slate-MDP*, as introduced by [Sunehag et al. 2015], wherein the recommender (i.e. the agent) interacts with the user (i.e. the environment) by recommending a slate of items (i.e. the action), followed by feedback from the user (i.e. the reward) (illustrated in Figure 1).

Related Work. The approach of constructing the recommendation task with slates instead of individual items is found across many research initiatives [Liu et al. 2021; Mehrotra et al. 2019; Zhao et al. 2018], which brings several modelling challenges to achieve effective generalisation performance, while dealing with the high combinatorics of slate recommendation. [Sunehag et al. 2015] proposed a setting in which slates are represented holistically and are considered *primitive actions*. The value of each slate is approximated using three DQN-based approaches, out of which two are capable of handling the combinatorics. However, they do not address the effectiveness of exploration and its impact on the long-term reward. Other initiatives in scientific literature adapt the slate recommendation scenario to bandit settings, without state transitions, and incorporates the notion of user preference variations through sudden changes in preference [Hariri et al. 2015]. This approach however disregards previous interaction session interaction history and primarily uses the current context to construct a recommendation. Furthermore, existing literature on user preference dynamics [Jiang et al. 2019] considers cases in which the user interest drift is described a non-linear stochastic function, an idea which we build on top of to integrate the the concept of session termination. Recent initiatives in deep learning have allowed for high-level feature extraction for modelling user preferences, which proved effective in recommending items that match recent interests [Tallapally et al. 2018]. Temporal models such as recurrent neural networks or hidden Markov models have been widely used to leverage the sequential user-item interactions of recommender systems, and have resulted in promising results estimating next-item predictions [Jannach and Ludewig 2017; Liu and Singh 2016; Mlika and Karoui 2020]. The former, suffers from scalability with large item and user spaces, and has limitations in capturing variational shifts over long horizons. The latter, on the other hand, fails to pick up the

Supervision
Responsible Professor: Dr. Frans A. Oliehoek[1]
Supervisors: Dr. Aleksander Czechowski[1], Davide Mambelli[1], Oussama Azizi[1].

Research Project in Computer Science and Engineering (CSE3000), TU Delft, EEMCS

user's timely interests. Advancements in Reinforcement Learning have allowed for the optimization of long-term value (LTV) in recommendations, while dealing with the large combinatorial action spaces, inherent to RL [Ie et al. 2019b]. However, such approaches encounter limitations in adapting to preference shifts over long-term horizons, particularly because of the assumed simplified user choice behaviour and generality in the user transition model.

Our Contribution. In this study, we seek to formalise the characteristics of dynamic user preferences, and use them to investigate the extent to which novel deep reinforcement learning algorithms perform under such dynamics. More explicitly, the aim of this is twofold. First, modelling the user preference trajectories and benchmarking the performance of two DQN-based algorithms, namely FullSlateQ and SlateQ, introduced by [Sunehag et al. 2015] and [Ie et al. 2019b], respectively. Second, observing the effect of long-term and immediate rewards on the algorithms' performance under the preference dynamics. Our contributions can be therefore broken down into two main sub-questions:

- **[Q1]** How do FullSlateQ and SlateQ compare against each other under dynamic user preferences? What factors drive their performance?
- **[Q2]** How does the tradeoff between long-term and immediate reward impact the value exploration in non-stationary environments?

The remainder of this paper is structured as follows. Section 2 introduces the problem formulation and relevant theoretical concepts for understanding this study. Subsequently, the 3 gives an in-depth overview of the core components that make up the simulation environment and the proposed framework for modelling the user preference dynamics, alongside relevant literature to support our approaches. The experimental setup and results are outlined in Section 4, followed by an analysis of the obtained findings in Section 5. Next, we reflect on the responsible research aspects of this work in Section 8. Lastly, we provide a summary and conclusion of this work coupled with future recommendations.

2 BACKGROUND

Before providing an analysis of the preference dynamics problem, we first describe the building blocks of our study. To this end, this section first introduces the modelling of the recommendation problem as a *slate-MDP*. Second, a brief description of the algorithms is provided. Last, we present the simulation environment used for training and evaluation.

2.1 Problem Formulation

We consider a setting in which a recommender agent interacts with an environment $\mathcal{E}$ (i.e. the user), by suggesting a list of items (i.e. *slates*) at each time step, out of which the user can either select one or no item (skips the selection). Once a document is selected, the user sends out a response to the selected item which consists of the engagement time. Following the response, the user can request subsequent recommendation slates, or terminate the session (i.e. the sequence of recommendations). The sequential nature of the sessions allows for modelling the recommendation task as a finite Markov decision process (MDP) with states $\mathcal{S}$, actions $\mathcal{A}$, transition probability $\mathcal{P}$, reward function $\mathcal{R}$ and discount factor γ (See Figure 1 for a simplified illustration). In this context, the 5-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ can be modelled as follows:

State space $\mathcal{S}$. Represents the user states $\mathcal{S} = \{s_1, s_2, ..., s_t\}$, which incorporate both static (fixed user attributes) and dynamic (user interests, time budget, and the interaction history at step s_t) features of the user.

Action space $\mathcal{A}$. Is a finite discrete space $\mathcal{A} = \{a_1, a_2, ..., a_t\}$, which represents the set of all possible slates to recommend at step t, based on the user state s_t. A single slate can be therefore formulated as follows $a_t = \{a_t^1, a_t^2, ..., a_t^K\}$, where K is the selected slate size.

Transition probability $\mathcal{P}$. Refers to the user state transition probability of moving to state s_{t+1} from state s_t, having taken action a_t, i.e. $p(s_{t+1} \mid s_t, a_t)$. It is worth noting that if the user selects no item, the user state still changes (non selection is still considered as user feedback), and thus $s_{t+1} \neq s_t$ holds, for all steps t.

Reward function $\mathcal{R}$. Having taken action a_t from state s_t, the agent receives the reward $\mathcal{R}(s_t, a_t)$ corresponding to the user feedback c_t (i.e. the user engagement over a recommendation session).

Discount factor γ. Represents the discount rate, indicating the value of future rewards. As γ approaches 0, the recommender agent behaves *myopically*, i.e. seeks to maximize immediate rewards, and when approaching 1, the agent gives a higher importance to future rewards.

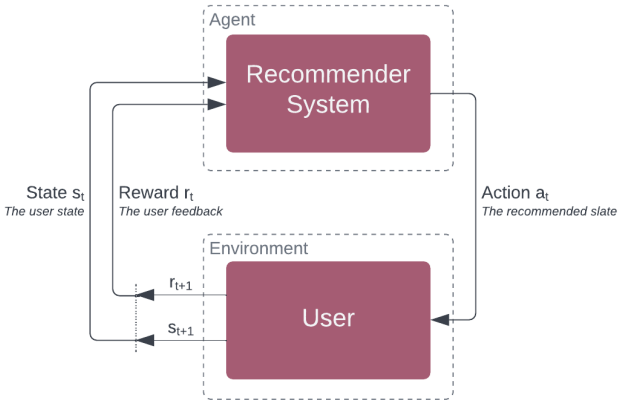

Figure 1: The Interactive Recommendation task modelled as a Reinforcement Learning problem

The aim of the recommender is to maximize the cumulative reward by learning a recommendation policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$. Given a deterministic policy π and the optimal policy π^*, the value function $V_\pi(s)$ and the optimal value function $V_\pi^*(s)$ can be defined as follows, respectively:

$$V_\pi(s) = \mathcal{R}(s, \pi(s)) + \gamma \cdot \sum_{s' \in \mathcal{S}} \mathcal{P}(s' \mid s, \pi(s)) \cdot V_\pi(s') \qquad (1)$$

$$V_\pi^*(s) = \max_\pi V_\pi(s) \tag{2}$$

The on-policy and optimal action-value functions can be defined analogously:

$$Q_\pi(s,a) = \mathcal{R}(s,a) + \gamma \cdot \sum_{s' \in \mathcal{S}} \mathcal{P}(s' \mid s,a) \cdot V_\pi(s') \tag{3}$$

$$Q_\pi^*(s,a) = \max_\pi Q_\pi(s,a) \tag{4}$$

In our work, the examined RL algorithms, namely FullSlateQ and SlateQ follow a DQN-based architecture, which improves generalisation in action-value function estimation by using a non-linear function approximator, a neural network, defined by [Mnih et al. 2013] as a *Q-network* with weights θ. More formally, $Q(s,a;\theta) \approx Q^*(s,a)$.

2.2 FullSlateQ

The FullSlateQ (FSQ) recommender agent implements full slate Q-learning based on a DQN Agent. The approach emplys a standard, non-decomposed, off-policy Q-learning algorithm, which treats each slate of items "holistically" (i.e. in its entirety) as a single action [Ie et al. 2019b]. As the authors point out, while the FSQ has a theoretical guarantee of convergence given enough exploration (for the interested reader, we recommend the proof of [Regehr and Ayoub 2021]), it is inherently unscalable, as it has to learn a large combinatorial space of Q-values. More specifically, given a document space $\mathcal{D}$ and a k-sized slate, the agent would have to keep track of $\binom{|\mathcal{D}|}{k} \cdot k!$ separate actions.

2.3 SlateQ

The large combinatorial action space of slate recommendation mentioned above poses several challenges that render the implementation of RL methods impractical in large-scale systems. One key difficulty is solving the combinatorial optimization problem of selecting a maximum Q-value slate (within the low-latency constraints of real-world systems), a necessary component for policy improvement. Therefore, [Ie et al. 2019b] introduce a decomposed version of the FullSlateQ algorithm, which allows for the decomposition of a slate Q-value into the individual Q-values of its items. The proposed decomposition framework circumvents the combinatorics associated with exploration and generalisation, and for action maximization, the authors describe a linear programming formulation of the *Long-term Value (LTV) slate optimization* problem, and provide two heuristics for tackling it, namely top-k and greedy.

2.4 RecSim

The deployment of novel recommender systems in real-world application poses several challenges to the researchers and practitioners in the field, predominantly centered around adequately modelling the interactive (and contextual) dynamics inherent to such applications [Kuanr and Mohapatra 2021].

To best observe the effects of such dynamics on recommendation performance, we turn to RecSim [Ie et al. 2019a], a configurable simulation environment, which allows for the study of RL in the context of stylized recommender settings. The RecSim environment has three main constituents, namely the *document model*, *user model* and *response model*. The recommender agent interacts with the environment by providing the user with a (fixed) slate of documents sampled from a known prior distribution P_D. The user then makes a selection via the configured *user-choice model* and sends its response to the recommender agent. Consequently, the user's state changes through a (configurable) transition function. For a visual interpretation of the data flow between the simulation components, we point the reader to Figure 1 of [Ie et al. 2019a]). It is worth mentioning that both the user and document models can have latent and observable features which can influence the user response to a recommended slate.

3 METHODOLOGY

Having established the building blocks in the previous section, we now introduce a detailed overview of the components that make up the RecSim simulation environment, and how they were constructed to depict an interactive dynamic environment. Subsequently, we provide the reader with the proposed preference dynamic framework for the user latent state transitions.

3.1 Simulation Components

To accurately benchmark the performance of the FullSlateQ and SlateQ algorithms within environments in which the latent (user) state undergoes gradual shifts, we turn to RecSim to develop a stylized model of the environment which reflects the dynamics of user behaviour. To this end, we provide an overview of core components of the environment.

3.1.1 ***Document Model.*** We consider a finite set of documents D, representing the candidate pool of items fit for recommendation. Recall from Section 2 that a single slate can be defined as $a_t = \{a_t^1, a_t^2, ..., a_t^K\}$, where K is the selected slate size. To abstract the notion of the time step t from a given item, we assume the document set to be independent of the interaction steps. Furthermore, we consider a finite set of topics T, which depicts all possible subjects a document can cover, and a user can be interested in. Each document $d \in D$ contains a *topic vector* $\delta \in [-1,\ 1]^{|T|}$, which denotes the degree of which the document reflects each of the topics, i.e. $\delta_i^{(d)}$ is the the degree to which document d reflects the topic i. The degree is ranging from -1 (totally uncorrelated) to 1, (extremely correlated).

To represent the versatility of topics in documents, each entry in the topic vector is distributed according to $\mathcal{N}(0,\ 0.5)$. The probability density function of the normally-distributed topic vector will serve an additional purpose in Subsection 3.2, for it will influence how a user's preference towards a given topic will shift over time.

Furthermore, each document d has a specific length λ_d, which could represent the length of a video, movie, song or article. The length will be used to measure the user engagement with a particular recommended item. In our work, the length is normally distributed, with $\mu_{length} = 5$ and $\sigma_{length} = 1$.

Finally, documents include a quality attribute q, representing a measure of overall attractiveness of a document of a given document. The quality of the documents is randomly sampled from a fixed Beta distribution $q \sim \mathcal{B}(\alpha_q, \beta_q)$ with $\alpha_q = 3$ and $\beta_q = 3$. The chosen probability distribution with the aforementioned parameter values is slightly skewed to the left, signifying a positive trend in quality across the document space. Each document's quality is sampled independently from the topic vector, and it is assumed to be user-independent.

3.1.2 ***User Model.*** User modelling plays a fundamental role in our investigation, as it will incorporate the dynamics of the environment. We consider a finite set of users $u \in U$ that interact with the recommender agent throughout the simulation. Each user u contains an *interest vector* μ, which similarly to the topic vector mentioned in the Document Model section, reflects the degree of interest towards a given topic for a user u. More formally, the interest vector can be defined as $\mu \in [-1,\ 1]^{|T|}$. Analogously to the document model, we assume the user's degree of interest towards a given topic to be independent of the other topics, and is normally distributed with mean 0 and standard deviation 0.5.

The user model receives two additional fixed parameters which control the probability of a preference shift p_{shift} and the step size ω at which the said shift occurs. Subsection 3.2 will offer a more in-depth treatment of the interplay between the user model parameters in the environment dynamics.

3.1.3 ***Response Model.*** Achieving high-quality recommendation slates that maximize the expected future reward is correlated with the user choice behaviour [Katahira 2015]. Choice modelling addresses the theory of individual decisions among a discrete set of options given stated or latent preferences within a particular context [Thill 2009]. Originally developed by economists and mathematical psychologists, user choice modelling has found numerous applications in fields such as econometrics, statistics and marketing [Van Cranenburgh et al. 2021]. In the context of this study, once the user is recommended a k-sized slate, they can observe the topic vector of each of the constituent document **before** making a choice. Only after the consumption of an document from the slate can the user observe the quality q of the selected document. We believe this to be a reasonable assumption, which is grounded in the design of common recommender scenarios (e.g. YouTube, Netflix).

For the choice model, we consider a multinomial proportional choice model, which assumes a finite set of user-item features x_{ij} for a user i and recommended document j in slate a. The features are computed by taking into account the dominant topic of the recommended document, and the user preference value of the corresponding topic, namely $x_{ij} = \mathbf{e}_{\arg\max \delta^{(j)}} \cdot \mu^{(i)}$, where $\mathbf{e}_k$ represents a one-hot encoded vector of the dominant topic (k). The choice model for a user i can be formulated as follows:

$$P(j \mid a) = \frac{x_{ij} - x_{norm}}{\sum_{k \in a} x_{ik} - x_{norm}} \tag{5}$$

Where $P(j \mid a)$ is the probability of choosing an item j, being presented a slate a, and $x_{norm} = -1$ is the normalisation constant to account for the negative interval $[-1,\ 0)$ of the user preference range.

3.2 User Preference Dynamics

Having analysed the core simulation components in the previous subsection, we now examine the theoretical underpinnings behind the dynamic user preference model. Recall from the problem formulation (Section 2.1), that we examine a setting in which the recommender interacts with a user at each time step t by suggesting a slate of items. To incorporate the sequential nature of the recommendation problem, we assume that at each time step t, the user's *interest vector* is resembled by μ_t, and $\mu_t^{(i)}$ is the user's interest in topic i at time t. Such formalisation will prove useful in reasoning about the user interest behaviour in subsequent parts of this section.

To investigate the extent to which recommender systems are capable of adapting to variational preference shifts, we first seek to formalise such dynamics in order to be well-aligned with real-life recommendation settings. One intuitive model of preference dynamics suggested by [Jiang et al. 2019], is that a user's interests fluctuate based on the interactions with the recommender system over time, that is $\mu_{t+1} > \mu_t$ for positive reinforcements (i.e. the user is more interested) and conversely, $\mu_{t+1} < \mu_t$ for negative reinforcements (i.e. the user is less interested), with a certain probability p_{shift} (See Section 3.1.2). We are interested in studying the dynamics of the user preference behaviour over extended horizons, and thus want to observe if the user interest vector μ_t at time step t, evolves from the initial set of preferences μ_0. The interest evolution problem can be formulated as follows:

$$\lim_{t \to \infty} \|(\mu_t - \mu_0)\|_2 > 0 \tag{6}$$

Where $\|\mathbf{x}\|_2 := \sqrt{x_1^2 + x_2^2 + ... + x_n^2}$ is the norm of a vector $\mathbf{x}$ in an n-dimensional Euclidean space $\mathbb{R}^n$.

Having introduced the notion of user preference shifts over long-term horizons, we now present the updating mechanisms employed in our experimentation. We first outline the evolution as a function of user interests, followed by the session termination model.

3.2.1 ***Interest Function-based Evolution.*** We construct a dynamic preference model by assuming a parameterized density function which describes the user interest distribution over a fixed interest range. In our experiments, we let the user interests follow a Gaussian distribution (See Section 3.1.2), with probability density function f, such that:

$$f(x;\ \mu,\ \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right) \tag{7}$$

We consider a gradient-based approach for determining the update magnitude of the user's interests towards a set of topics. It is worth noting the fact that the probability density function illustrated in Equation 7 is defined and differentiable in the user preference range $[-1,\ 1]$ described in section 3.1.2. Taking the first derivative (Equation 8) will yield the gradient of the density function describing the user interests, which can be interpreted as

the user's *inclination towards preference change*. This approach is quite similar to [Jiang et al. 2019], where the authors model the user preference *drift* $\mu_{t+1} - \mu_t$ as a nonlinear stochastic function. The discrepancy however, is that we use the function describing a user's interest as a proxy for determining the direction (positive or negative) of the preference change, and the magnitude of the update.

$$\nabla f(x;\ \mu,\ \sigma) = -f(x;\ \mu,\ \sigma) \cdot \left(\frac{x-\mu}{\sigma^2}\right) \tag{8}$$

We propose the following gradient-based algorithm for stochastically updating the user's interest, based on the items consumed.

Algorithm 1 Gradient-based Preference Updating

Require: $\mu_t, \delta, \mu_{interest}, \sigma_{interest}, p_{shift}, \omega, n$

```
mask ← Vector in [0,1]^|T| containing n random 1-positions
target_interests ← mask · μ_t
target_topics ← mask · δ
Estimate ∇f = ∇f(target_interests; μ_interest, σ_interest)
Compute Δ_t(μ_t) = |target_topics · ∇f|      ▷ Absolute change in user interest
Perform an update with probability p_shift:
    p(update, positive) ← ⌊(μ_t+1)/(n+1) · mask⌋
    With probability p(update, positive):
        μ_t+1 ← μ_t + ωΔ_t(μ_t)              ▷ Positive update
    With probability 1 − p(update, positive):
        μ_t+1 ← μ_t − ωΔ_t(μ_t)              ▷ Negative update
```

Let μ_t be the user interest vector prior to consumption of a particular document $d \in D$ with an associated topic vector δ. In Algorithm 1, n depicts the number of entries in the user interest vector that will be updated. In our experiments, we set $n = 3$. A positive update, $\mu_{t+1} \leftarrow \mu_t + \omega\Delta_t(\mu_t)$, occurs with probability $\left\lfloor \frac{\mu_t+1}{n+1} \cdot mask \right\rfloor$, while a negative shift, $\mu_{t+1} \leftarrow \mu_t - \omega\Delta_t(\mu_t)$, occurs with probability $1 - \left\lfloor \frac{\mu_t+1}{n+1} \cdot mask \right\rfloor$. Hence, stronger interests are more likely to be reinforced, while weaker interests have a higher likelihood of being diminished. A visualisation of the mechanism is illustrated under Figure 2.

3.2.2 ***Session Termination.*** The second approach taken towards modelling the user preference dynamics is based on session termination, a concept adapted from [Ie et al. 2019b]. We assume that every user has an initial time budget B_u at the start of each recommendation session, unobservable to the recommender agent. Once a user u engages with a document d, the engagement time $e_u^{(d)}$ is subtracted from the time budget. Depending on the quality of the document, which is observed by the user after consuming it, partly replenishes the user's time budget. Thus higher quality documents encourage the user to engage with more content throughout a recommendation session, which acts as the reward of this model (i.e. the feedback of the user).

The session termination model is constructed as follows. The replenishment of the time budget is centered around three components, namely the user engagement, the perceived utility, and

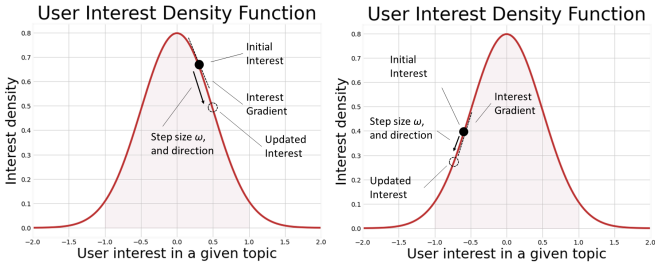

Figure 2: Illustration of the Interest Function-based updating approach for a given topic. The figure on the left resembles a step taken in the "positive" direction, i.e. the user's interest in the topic is reinforced. The figure on the right represents a step towards the "negative" direction, namely the user's interest in the topic is decreased.

a retention factor $\phi \sim U(0.1,\ 0.2)$. We assume the user model to additionally contain a quality factor $\rho_u \sim \beta(9,\ 3)$, which indicates the affinity of a sampled user towards higher-quality content. Using the Beta-distributed quality factors indicate a somewhat more pronounced appeal towards higher quality items in recommendation; weighing in the quality items has been shown to improve recommendation accuracy [Cho et al. 2008]. The user's time budget dynamics can be therefore defined as:

$$B_u^{(t+1)} = B_u^{(t)} - e_u^{(d)} + \phi \cdot e_u^{(d)} \cdot u(u,d) \tag{9}$$

Where the perceived utility is computed using the document and item quality attributes:

$$u(u,d) = \rho_u \cdot q_d \tag{10}$$

The intuition behind session termination is that consuming documents of higher quality has a more positive impact on the cumulative user engagement, i.e decrease the budget at a slower rate than lower quality documents. This approach can potentially test if the recommender agent can adapt to suggest higher quality items that match with the user's unobservable affinity ρ_u towards higher quality documents.

3.3 User Engagement and Satisfaction

Having covered the two approaches used for constructing a non-stationary environment in the previous subsection, we now present two models used for calculating the user's engagement with a recommended item, and subsequently the internal satisfaction received from consuming the document.

Once the user selects the document to consume from the recommendation slate, the engagement time $e_u^{(d)}$ is computed based on the similarity between the user's interest vector μ_t and the item's topic vector δ using the cosine similarity (Equation 11), a common metric used in the context of recommender systems [Benard Magara et al. 2018] .

$$\cos\varphi = \frac{\mu_t \cdot \delta}{\|\mu_t\|\ \|\delta\|} \tag{11}$$

In Sections 3.1.2 and 3.1.1, we assume both the interest and topic vector to be vary independently according to a normal distribution. From initial empirical evaluations, we have observed that the similarity values of these vectors do not attain values close to the cosine boundaries $[-1,\ 1]$, but more in the interval $[-0.7,\ 0.7]$. Therefore we account for the shortened interval by applying a sigmoid function to the cosine similarity with a sensitivity parameter $\tau = 7$:

$$e_u^{(d)} = \sigma(\tau \cdot cos\varphi) \cdot \lambda_d = \frac{\lambda_d}{1 + e^{-\tau \cdot cos\varphi}} \tag{12}$$

The user's satisfaction sat_t varies throughout the recommendation session. We define the satisfaction in two parts - Equations 13 and 14:

$$sat = \alpha \cdot sat_t + \beta \cdot cos\varphi + \mathcal{N}(0, \eta) \tag{13}$$

Where α and β represent user-specific memory (forgetfulness) and immediate discounts, respectively. In our experiments, α is set to 0.7, and $beta$ to 1.0, signifying that the user satisfaction is predominately determined by the match between the user's interest and the recommended item. Finally, we add some noise in the satisfaction with parameter $\eta = 0.03$. Similarly to the user engagement, we pass the calculated satisfaction through a sigmoid function adapted to the empirically observed values of the satisfaction $[-0.4,\ 0.4]$, with sensitivity parameter $\tau = 3$.

$$sat_{t+1} = \sigma(\tau \cdot sat) = \frac{1}{1 + e^{-\tau \cdot sat}} \tag{14}$$

4 EXPERIMENTS

We present the experimental results from the comparison of the two deep reinforcement leaning agents discussed in Section 2, namely FullSlateQ and SlateQ, under a range of parameterized non-stationary environments, which support different levels of magnitude in their dynamics. To justify the validity of our findings, we subsequently perform the test on two other baseline recommender agents, a standard Q-learning algorithm and a Random agent. In this section we first introduce the setup of our experiments, followed by an analysis of the user preference dynamics. Finally, we discuss the experimental results and their significance in answering the research questions.

4.1 Experimental setup

We now outline the experimental setup for training and evaluating the agents within a dynamic environment. The main task in our experiments is maximizing the user's engagement throughout a recommendation session, and it's overall satisfaction at the end of the session. The RL algorithms used, namely FullSlateQ, SlateQ and Q-learning, are implemented in the RecSim simulated environment using Dopamine [Castro et al. 2018]. To assess the performance of the algorithms, we keep track of both user satisfaction and overall engagement, as described in Section 3.3, and we observe the quality and satisfaction attributes of the recommended items.

We consider the each user's initial budget $B_u = 500$ time units, each document d, with length $\lambda_d \sim \mathcal{N}(5, 1)$. If no document is clicked, a penalty of 1 time unit is set, and the budget is updated according to Equation 12. We set the number of items to 10, the slate size to 2 and the number of topics $|T|$ to 20. We evaluate the FullSlateQ and SlateQ algorithms by additionally comparing them alongside a tabular implementation of Q-learning and a random policy. We perform each experiment using three randomly generated seeds, and take the average of the performance to obtain a more robust measure of the algorithms' performance.

4.2 Preference Dynamics Analysis

As presented in Section 8, we propose a user preference updating mechanism based on a density function. Figures 3 and 4 depict a user's interest evolution across 7k training steps.

Under mild conditions (Figure 3), the user's interest barely fluctuates, and behaves almost stationary for low values of p_{shift} (0.001, 0.005, 0.05). On the other hand, when p_{shift} takes higher values (0.1, 0.5), the preference shifts become more apparent. When varying the step size ω, we also observe preference changes of higher magnitude (e.g. Figure 4, $p_{shift} = 0.5$, $\omega = 1.0$). One potential issue that can be identified from both graphs is that converging to the interest interval extremes (-1 and 1), might be difficult to combat due to the low gradient values in that vicinity.

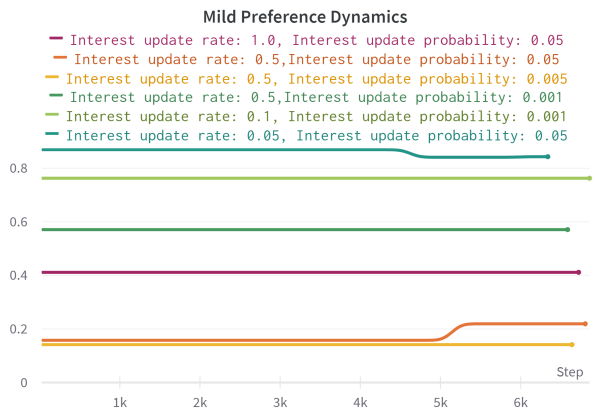

Figure 3: The user preference value towards a topic t, under mild preference updating conditions, across one iteration

4.3 Results

Within this section, we describe our empirical findings, and briefly outline their significance towards answering the research questions. All results showcased here are within a 95% confidence interval. We first outline the algorithms' performance under stationary user preferences, and then investigate the recommendation performance under preference dynamics. Further experimental results alongside more documentation on the hyperparameters of the environment will be made accessible in the code repository.

We initially attempt to replicate the findings of [Ie et al. 2019b] in order to test the quality the recommendation policies using SlateQ, under our preference dynamic conditions. Similar results were achieved for non-myopic long-term value policies that optimize for user engagement, yet the quality of recommendations didn't show improvements, and converges to around 0.66, as seen in Tables 1,

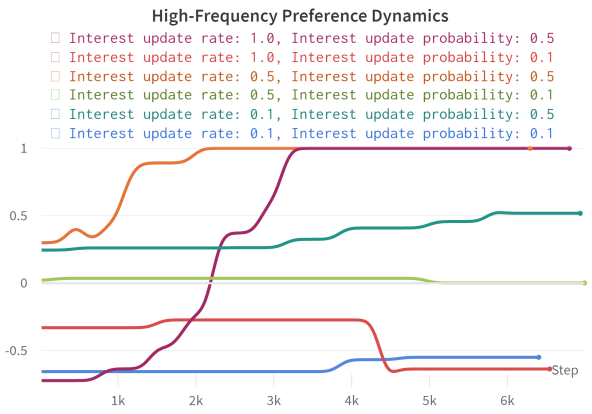

Figure 4: The user preference value towards a topic t, under high-frequency preference updating conditions, across one iteration

2 and 3. This is partly because user engagement is also driven by similarity between the user interests and the topic vector of the document, and not just the quality, alongside the fixed discounting factor $\gamma = 0.5$.

Stationary User Preferences			
Algorithm	Avg. Rewards	Avg. Quality	Avg. Sat.
Random	424.2	0.66	0.8307
Q-Learning	424.6 (+0.09%)	0.66 (+0%)	0.8412 (+1.26%)
FullSlateQ	424.58 (+0.08%)	0.67 (+1.51%)	0.8409 (+1.22%)
SlateQ	463.23 (+9.2%)	0.68 (+3.03%)	0.873 (+5.1%)

Table 1: The performance of the algorithms under stationary user preference conditions: $p_{shift} = 0$.

To adequately evaluate the performance of the FullSlateQ and SlateQ algorithms in dynamic environments, we first investigate their performance under a stationary environment: $p_{shift} = 0$. We present our results in 1. We train each algorithm over 12k training steps, and evaluate the performance over 3000 different users in a hold out evaluation set. We also display the relative performance improvement to the random baseline next to the metrics. While the average reward over episodes of FullSlateQ does not show drastic improvement over the two baselines, SlateQ yields a moderate increase in performance of 9%. Additionally, SlateQ is able to offer a 5.1% improvement over the baseline and achieves the highest user satisfaction out of all the experiments.

To evaluate the impact of non-stationary environments on the algorithms' performance, we run multiple experiments with different user preference conditions ($p_{shift} = 0.001, 0.005, 0.05, 0.1, 0.5$ and $\omega = 0.05, 0.1, 0.5$). The results of all the experiments can be found in the accompanying repository containing the source code, yet here in this work we present the most representative results. To this end, we divide the problem into two cases: (i) a mild preference dynamics setting, where $p_{shift} = 0.005, \omega = 0.05$ and (ii) a high-frequency preference dynamic setting, with $p_{shift} = 0.1, \omega = 0.5$,

Mild User Preference Dynamics			
Algorithm	Avg. Rewards	Avg. Quality	Avg. Sat.
Random	421.27	0.664	0.841
Q-Learning	423.17 (+0.45%)	0.6667 (+0.41%)	0.841 (+0%)
FullSlateQ	424.72 (+0.82%)	0.667 (+0.45%)	0.841 (+0%)
SlateQ	465.25 (+10.43%)	0.68 (+2.41%)	0.854 (+0.95%)

Table 2: The performance of the algorithms under mild user preference conditions: $p_{shift} = 0.005, \omega = 0.05$.

High-Frequency Preference Dynamics			
Algorithm	Avg. Rewards	Avg. Quality	Avg. Sat.
Random	420.11	0.64	0.841
Q-Learning	424.74 (+1.1%)	0.667 (+4.06%)	0.842 (+0.16%)
FullSlateQ	424.35 (+1.01%)	0.667 (+4.32%)	0.842 (+0.14%)
SlateQ	479.8 (+14.22%)	0.668 (+4.375%)	0.857 (+1.91%)

Table 3: The performance of the algorithms under high-frequency user preference conditions: $p_{shift} = 0.1, \omega = 0.5$.

and showcase their performance in Table 2 and Table 3, respectively. Similarly to the stationary environment, the average quality remains centered at around 0.67 for all agents, indicating that the agents are unable to converge to higher-quality documents.

Remarkably, under both mild and high-frequency conditions, SlateQ offers higher overall user engagement than the stationary setting, with 10.43% and 14.22% improvements over the Random agent, respectively. This is most likely attributed to the fact that the slate decomposition reduced the complexity of generalisation and allows for more effective TD and Q-learning [Ie et al. 2019b].

5 DISCUSSION

We now present a more in-depth treatment of the results outlined in Section 4, and correlate them with answering our both our research questions.

5.1 Comparison between FullSlateQ and SlateQ

In order to approach the first research question **RQ1**, we measure the performance of the algorithms by observing the average reward (i.e. user engagement) over recommendation sessions. We show the results in Table 2 and Table 3, and illustrate the performance of FullSlateQ and SlateQ in Figure 5.

Our findings suggest that SlateQ performs significantly better than FullSlateQ and the two baselines, offering, on average, a near 10.57% improvement over FullSlateQ. Furthermore, SlateQ continues to outperform FullSlateQ on user satisfaction (See Figure 6) and recommendation quality, although not as apparent for the latter. Additionally, FullSlateQ has to learn the Q-values of $\binom{10}{2} \cdot 2! = 90$ distinct slates, which makes exploration and generalisation much more difficult in reasonable time. Moreover, the training time of FullSlateQ takes around 1.5X more time then that of SlateQ for this particular slate configuration, yet previous experiments with

$k = 3$ slate, i.e. $\binom{10}{3} \cdot 3! = 720$ states, the training time equaled to around 6X that of SlateQ. The aforementioned findings indicate the substantial value in Slate decomposition approaches, as they not only exhibit increased performance over FullSlateQ, but also render RL tractable with slate recommendations.

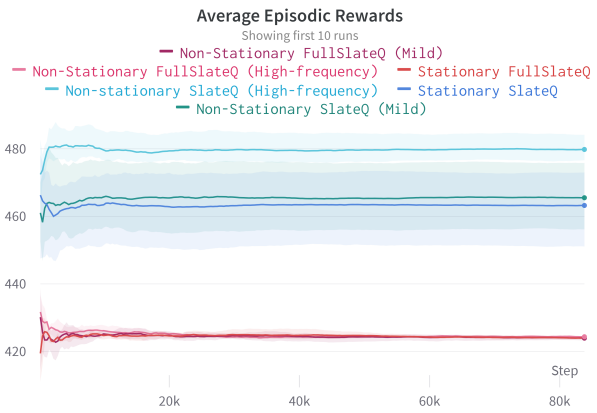

Figure 5: The average episodic rewards of FullSlateQ and SlateQ in evaluation for stationary and non-stationary environments.

5.2 Effect of long-term and immediate reward on value exploration

To investigate the impact of immediate and long-term value policies over the value exploration - **RQ2**, we turn our focus towards the average quality of recommended items and the user satisfaction. We perform the experiments with a discount factor $\gamma = 0.5$, and we observe that the the agents generally adopt a *myopic* policy when recommending items, optimizing for immediate reward. In our case translates to suggesting items with topics closer to the user's interest, resulting in higher satisfaction (as seen in Figure 6), yet lower values for quality. Thus, the agents fail to make an suitable tradeoff between guiding the user's preferences towards higher quality documents at the expense of temporarily diminishing the user budget.

6 CONCLUSION AND FUTURE WORK

In this study we have formulated the recommendation problem as a *slate-MDP*, and have investigated the ability of FullSlateQ and SlateQ to learn recommendation policies through sequential interactions over extended horizons in non-stationary environments. Based on the results obtained using RecSim, we can conclude that SlateQ offers notable improvements (10.57%) in user engagement compared to FullSlateQ in dynamic environments, while concomitantly rendering RL tractable with slates, and therefore scalable in practical applications. On the other hand, we have observed how with a relatively moderate discounting factor, both SlateQ and FullSlateQ generally adopt a myopic recommendation policy, and therefore failing to make a suitable tradeoff to maximize recommendation quality and user engagement.

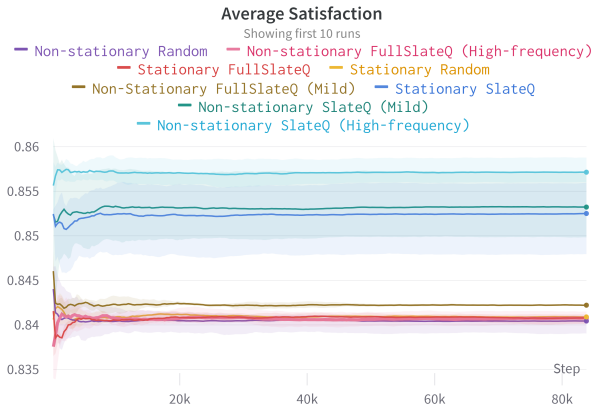

Figure 6: The average Satisfaction of FullSlateQ, SlateQ and Random agents in evaluation for stationary and non-stationary environments.

Our contributions suffer from two limitations. First, to reliably measure the impact preference dynamics on the recommender systems, a more accurate non-linear model could be implemented to resemble the nature of user change in interest. Second, the synthetic nature of the stylized environment carries several key assumptions about the user and item spaces. In our case, one particular example of such assumption would be the interdependence of interest between topics. Thus, in future work, online testing should be carried out to validate the findings in a real-world setting. Finally, further experimentation with different values of discounting factors would prove useful in evaluating if optimizing for long-term user engagement using DRL yields performance increase over myopic approaches.

7 ACKNOWLEDGEMENTS

The computational resources that aided the experimentation were provided by the Delft High Performance Centre [Delft High Performance Computing Centre (DHPC) 2022].

8 RESPONSIBLE RESEARCH

Throughout the study, we sought to adhere to the principles of conducting reproducible and ethical computational research. This section first reflects on the ethical implications of this work. We then address the reproducibility of the work for potential future contributions.

8.1 Ethical Implications

Recommender Systems are ubiquitous and have been widely adopted by several digital services ranging from streaming services to e-commerce.

Gathering user personal information often poses serious privacy risks and challenges. These impasses may be seen as unavoidable, given that the majority of successful recommender systems rely heavily on user data to create personalised recommendations. We approach this risk by using synthesized user data, sampled from a fixed distribution with injected probabilistic noise, as described in

Section 3.1.2. This helps us achieve two of our desiderata: eliminating privacy risks, and modelling the sequential, interactive nature of a recommendation system. Notwithstanding, conclusions regarding the performance of RS algorithms under various stylized contexts with synthetic data should be interpreted carefully, as further research using real-life data is necessary to validate their performance before migrating to commercial use.

8.2 On the Reproducibility of Experiments

The reproducibility of experiments is an important aspect of Machine Learning research, as it allows for verifying the reliability and integrity of the proposed results. To ensure that the experimental results from this work can be reproduced, we provide all the necessary resources and abide by the principles stated in the Machine Learning Reproducibility checklist, introduced in the 2019 edition of the Neural Information Processing Systems (NeurIPS) conference [Pineau et al. 2020]. We further make the source code and supplementary material related to experimentation publicly accessible[1].

REFERENCES

G. Adomavicius and Jingjing Zhang. 2012. Stability of recommendation algorithms. *ACM Trans. Inf. Syst. (TOIS)* 30 (01 2012).

Charu C. Aggarwal. 2016. *An Introduction to Recommender Systems.* Springer International Publishing, Cham, 1–28. https://doi.org/10.1007/978-3-319-29659-3_1

Maake Benard Magara, Sunday O. Ojo, and Tranos Zuva. 2018. A comparative analysis of text similarity measures and algorithms in research paper recommender systems. In *2018 Conference on Information Communications Technology and Society (ICTAS).* 1–5. https://doi.org/10.1109/ICTAS.2018.8368766

Pablo Samuel Castro, Subhodeep Moitra, Carles Gelada, Saurabh Kumar, and Marc G. Bellemare. 2018. Dopamine: A Research Framework for Deep Reinforcement Learning. https://doi.org/10.48550/ARXIV.1812.06110

Sunghoon Cho, Moohun Lee, Jeongseok Kim, Bonghoi Kim, and Euiin Choi. 2008. A Novel Weight for Recommendation: Item Quality. In *2008 International Conference on Computational Sciences and Its Applications.* 39–46. https://doi.org/10.1109/ICCSA.2008.58

Delft High Performance Computing Centre (DHPC). 2022. DelftBlue Supercomputer (Phase 1). https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1.

Negar Hariri, Bamshad Mobasher, and Robin Burke. 2015. Adapting to User Preference Changes in Interactive Recommendation. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15).* AAAI Press, 4268–4274.

Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *2008 Eighth IEEE International Conference on Data Mining.* 263–272. https://doi.org/10.1109/ICDM.2008.22

Liwei Huang, Mingsheng Fu, Fan Li, Hong Qu, Yangjun Liu, and Wenyu Chen. 2021. A deep reinforcement learning based long-term recommender system. *Knowledge-Based Systems* 213 (2021), 106706. https://doi.org/10.1016/j.knosys.2020.106706

Eugene Ie, Chih-Wei Hsu, Martin Mladenov, Vihan Jain, Sanmit Narvekar, Jing Wang, Rui Wu, and Craig Boutilier. 2019a. RecSim: A Configurable Simulation Platform for Recommender Systems. *CoRR* abs/1909.04847 (2019). arXiv:1909.04847 http://arxiv.org/abs/1909.04847

Eugene Ie, Vihan Jain, Jing Wang, Sanmit Narvekar, Ritesh Agarwal, Rui Wu, Heng-Tze Cheng, Morgane Lustman, Vince Gatto, Paul Covington, Jim McFadden, Tushar Chandra, and Craig Boutilier. 2019b. Reinforcement Learning for Slate-based Recommender Systems: A Tractable Decomposition and Practical Methodology. *CoRR* abs/1905.12767 (2019). arXiv:1905.12767 http://arxiv.org/abs/1905.12767

F.O. Isinkaye, Y.O. Folajimi, and B.A. Ojokoh. 2015. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal* 16, 3 (2015), 261–273. https://doi.org/10.1016/j.eij.2015.06.005

Dietmar Jannach and Malte Ludewig. 2017. When Recurrent Neural Networks Meet the Neighborhood for Session-Based Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17).* Association for Computing Machinery, New York, NY, USA, 306–310. https://doi.org/10.1145/3109859.3109872

Gawesh Jawaheer, Peter Weller, and Patty Kostkova. 2014. Modeling User Preferences in Recommender Systems: A Classification Framework for Explicit and Implicit User Feedback. *ACM Trans. Interact. Intell. Syst.* 4, 2, Article 8 (jun 2014), 26 pages. https://doi.org/10.1145/2512208

Ray Jiang, Silvia Chiappa, Tor Lattimore, András György, and Pushmeet Kohli. 2019. Degenerate Feedback Loops in Recommender Systems. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society.* ACM. https://doi.org/10.1145/3306618.3314288

Kentaro Katahira. 2015. The relation between reinforcement learning parameters and the influence of reinforcement history on choice behavior. *Journal of Mathematical Psychology* 66 (2015), 59–69. https://doi.org/10.1016/j.jmp.2015.03.006

Pigi Kouki, Shobeir Fakhraei, James Foulds, Magdalini Eirinaki, and Lise Getoor. 2015. HyPER: A Flexible and Extensible Probabilistic Framework for Hybrid Recommender Systems. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15).* Association for Computing Machinery, New York, NY, USA, 99–106. https://doi.org/10.1145/2792838.2800175

Madhusree Kuanr and Puspanjali Mohapatra. 2021. *Recent Challenges in Recommender Systems: A Survey.* 353–365. https://doi.org/10.1007/978-981-15-6353-9_32

David Zhan Liu and Gurbir Singh. 2016. A recurrent neural network based recommendation system. In *International Conference on Recent Trends in Engineering, Science & Technology.*

Shuchang Liu, Fei Sun, Yingqiang Ge, Changhua Pei, and Yongfeng Zhang. 2021. Variation Control and Evaluation for Generative Slate Recommendations. In *Proceedings of the Web Conference 2021 (WWW '21).* Association for Computing Machinery, New York, NY, USA, 436–448. https://doi.org/10.1145/3442381.3449864

Linyuan Lu, Matus Medo, Chi Ho Yeung, Yi-Cheng Zhang, Zi-Ke Zhang, and Tao Zhou. 2012. Recommender systems. *Physics Reports* 519, 1 (2012), 1–49. https://doi.org/10.1016/j.physrep.2012.02.006 Recommender Systems.

Rishabh Mehrotra, Mounia Lalmas, Doug Kenney, Thomas Lim-Meng, and Golli Hashemian. 2019. Jointly Leveraging Intent and Interaction Signals to Predict User Satisfaction with Slate Recommendations. In *The World Wide Web Conference (WWW '19).* Association for Computing Machinery, New York, NY, USA, 1256–1267. https://doi.org/10.1145/3308558.3313613

Fatma Mlika and Wafa Karoui. 2020. Proposed Model to Intelligent Recommendation System based on Markov Chains and Grouping of Genres. *Procedia Computer Science* 176 (2020), 868–877. https://doi.org/10.1016/j.procs.2020.09.082 Knowledge-Based and Intelligent Information Engineering Systems: Proceedings of the 24th International Conference KES2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. *CoRR* abs/1312.5602 (2013). arXiv:1312.5602 http://arxiv.org/abs/1312.5602

Isshu Munemasa, Yuta Tomomatsu, Kunioki Hayashi, and Tomohiro Takagi. 2018. Deep reinforcement learning for recommender systems. In *2018 International Conference on Information and Communications Technology (ICOIACT).* 226–233. https://doi.org/10.1109/ICOIACT.2018.8350761

Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Hugo Larochelle. 2020. Improving Reproducibility in Machine Learning Research (A Report from the NeurIPS 2019 Reproducibility Program). https://doi.org/10.48550/ARXIV.2003.12206

Matthew T. Regehr and Alex Ayoub. 2021. An Elementary Proof that Q-learning Converges Almost Surely. *CoRR* abs/2108.02827 (2021). arXiv:2108.02827 https://arxiv.org/abs/2108.02827

Peter Sunehag, Richard Evans, Gabriel Dulac-Arnold, Yori Zwols, Daniel Visentin, and Ben Coppin. 2015. Deep Reinforcement Learning with Attention for Slate Markov Decision Processes with High-Dimensional States and Actions. *CoRR* abs/1512.01124 (2015). arXiv:1512.01124 http://arxiv.org/abs/1512.01124

Dharahas Tallapally, Rama Syamala Sreepada, Bidyut Kr. Patra, and Korra Sathya Babu. 2018. User Preference Learning in Multi-Criteria Recommendations Using Stacked Auto Encoders. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18).* Association for Computing Machinery, New York, NY, USA, 475–479. https://doi.org/10.1145/3240323.3240412

J.-C. Thill. 2009. Choice Modeling. In *International Encyclopedia of Human Geography*, Rob Kitchin and Nigel Thrift (Eds.). Elsevier, Oxford, 78–83. https://doi.org/10.1016/B978-008044910-4.00413-2

S. Van Cranenburgh, S. Wang, A. Vij, F. Pereira, and J. Walker. 2021. Choice modelling in the age of machine learning – discussion paper. https://doi.org/10.48550/ARXIV.2101.11948

Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Dawei Yin, Yihong Zhao, and Jiliang Tang. 2018. Deep Reinforcement Learning for List-wise Recommendations. *CoRR* abs/1801.00209 (2018). arXiv:1801.00209 http://arxiv.org/abs/1801.00209

[1] https://github.com/lucapantea/research-project