Building Healthy Recommendation Sequences for Everyone: A Safe Reinforcement Learning Approach

Ashudeep Singh*
ashudeep@cs.cornell.edu
Cornell University

Yoni Halpern, Nithum Thain, Konstantina Christakopoulou, Ed H. Chi, Jilin Chen, Alex Beutel
Google Research

ABSTRACT

A key consideration in the design of recommender systems is the long term well-being of users. In this work, we formalize this challenge as a multi-objective safe reinforcement learning problem, balancing positive user feedback and the "healthiness" of user trajectories. We note that in some cases, naively balancing these objectives can lead to unhealthy experiences, even if unlikely, still occurring in a small subset of users. Therefore, we examine a distributional notion of recommendation safety. We propose a reinforcement learning approach that optimizes for positive feedback from users while simultaneously optimizing for the health of worst-case users to remain high. To empirically validate our method, we develop a research simulation environment motivated by a MovieLens recommendation setting that considers exposure to violence as a proxy for unhealthy recommendations. We demonstrate that our method reduces unhealthy recommendations to the most vulnerable users without sacrificing much user satisfaction.

1 INTRODUCTION

Recommendations play an important role in the choices people make in many areas including entertainment, shopping, food, news, employment, and education. These recommender systems are typically optimized for user engagement or satisfaction [25, 44], but it is also valuable in many applications for them to create positive and healthy user experiences [31]. For example, users may want repeated recommendations for restaurants to both be enjoyable to users and also mostly healthy [3]. Recommending *only* ice cream, while enjoyable, is likely not a healthy experience in the longer term. Similarly in movie recommendation, some users may prefer a recommender system that recommends enjoyable movies but limits the amount of exposure to violence or some other negative content [1, 13].

How should a recommender responsibly make item recommendations when selecting from a corpus that contains some amount of potentially unhealthy content like violence in movies or poor nutrition in restaurants? Both nutrition and amount of violence are examples of attributes that can be represented as continuous quantities, and while a positive user experience may include the occasional interaction with these types of items, repeated exposure may ultimately lead to a negative user experience. Hence, in formulating the problem, we examine two competing objectives: optimizing for short-term positive feedback, like ratings, and limiting long-term exposure to unhealthy items. As we are concerned with *long-term* user enjoyment and health, the problem lends itself well to reinforcement learning, which researchers have shown in recent years to be an effective framework for recommender systems to improve long-term user satisfaction [9, 59, 63]. For long-term healthiness, we may not be concerned if individual recommendations are unhealthy but the goal is to limit the overall exposure to unhealthy recommendations over the course of the user experience; we propose this as an additional reward signal.

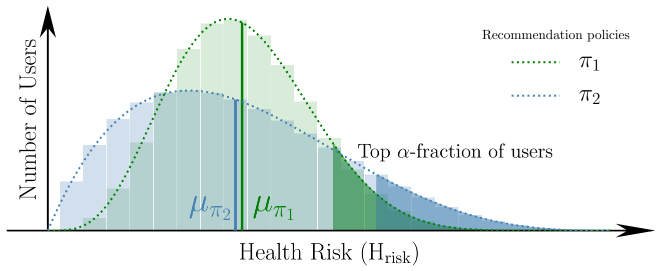

Figure 1: Recommendation policies π_1 and π_2 have similar mean risk but they pose significantly different risks to the worst-case users.

A natural way to balance these two objectives is through a multi-task recommendation approach, where for each user these two long-term benefits are factored into recommendation choices, but we find this approach to be incomplete for our goal of responsible recommendation. In any recommender system, some users are more likely to experience unhealthy trajectories than others. For example, users who enjoy movies in the crime genre might be exposed to violence more often than users interested in the comedy genre. A recommender that naively frames this as a multi-task problem would optimize for the average user experience, but could still result in some (small) fraction of users with substantially unhealthy experiences (Figure 1). Rather, we would prefer a recommender system that doesn't just improve the healthiness of user experiences on average, but also improves the healthiness of the least-healthy experiences.

A normative goal of our work is to satisfy the Rawlsian principle of maximin welfare for distributive justice [42]. According to the maximin principle, a fair system should be designed to maximize the position of those who will be worst-off in it. To develop a recommendation approach that maximizes the "healthiness" of worst-case user experiences, we propose a distributional notion of "Health Risk" (denoted H_{risk}) that focuses on the worst-case user experiences as compared to the average. We then incorporate this notion into our objective function for learning a sequential recommendation policy that optimizes an engagement or satisfaction

*Work conducted while the author was an intern at Google Research. Correspondence address: ashudeep@cs.cornell.edu metric for all the users while limiting H_{risk} for the worst-off cases. We propose a policy gradient algorithm to train a Reinforcement Learning agent on this objective that adapts to the user's behavior in a sequential recommendation setting.

Contributions: In this work, we take a step towards building responsible recommender systems that aim to create positive and healthy user experiences, even for the worst-case users. We make the following contributions in the rest of the paper:

- **Problem Framing:** We propose a novel definition for safe reinforcement learning in recommender systems (Section 3) where safety is defined as an exposure metric (*health risk*) for worst-case user trajectories.
- **Safe RL Recommendation Algorithm:** We show how to optimize this metric in a sequential recommendation setting with a policy gradient algorithm (Section 4).
- **Responsible Recommendation Simulation Environment:** To evaluate the effectiveness of our algorithm, we formulate a simulation environment, using the MovieLens dataset [22], where we consider violence as a problematic feature limit exposure to in the long-term, especially for the users whose preferences are highly correlated with violence e.g. users who like crime or action films (Section 5).
- **Empirical Benefits:** Finally, we validate our method by showing how the risks are distributed across different users, and how our method optimizes both reward and healthiness together. We demonstrate that optimizing for the worst-case user trajectories leads to superior tradeoffs of rating against the health of the average and the worst-case users (Section 6).

2 RELATED WORK

Conventionally, recommender systems (RS) have used collaborative filtering algorithms to build an accurate model of short-term feedback from user history [20, 43]. For this task, latent factor models and matrix factorization techniques have shown promise in many real-world rating prediction tasks [32]. Recently there has been a growing interest in studying sequential interactive recommender systems that treat recommendations as a sequence of interactions with users [23]. Given the considerable success of Reinforcement Learning (RL) in games [40], robotics [7], and physical system control [52], it has become a common framework to train recommenders that optimize user feedback over the entire sequence [10, 59, 64].

However, the use of RL for recommendations brings new challenges of its own. Since data collection for recommender systems in practice requires interaction with real users, it is often necessary to use offline data from past recommendation policies to train better ones [9]. Another challenge is the size of the action space that involves choosing from a large vocabulary ranging from hundreds to millions depending on the recommendation task [14, 29].

In this work, we will adopt the Safe Reinforcement Learning (Safe RL) approach to building recommender systems. The problem of safety in RL has been a long-studied topic, originating in robotics where the agent must avoid physical damage while completing tasks [37]. Additionally, safety has been studied in other contexts where risk is formulated as a function of different sources of uncertainties in the environment and agent interactions e.g. the stochasticity of the agent [12, 24], variance of the rewards [46], worst-case outcome [6], probability of an agent producing a high-risk trajectory [37] or reaching an unsafe state [18] (see García et al. [17] for a comprehensive survey). In this work, we specifically use the percentile risk criteria defined on a measure that we will refer to as the health risk of the recommendation trajectories (Section 3) and adapt the Policy Gradient algorithm from Tamar et al. [54] to train recommendation agents (Section 4).

Our work is motivated by the growing community of research in fairness, accountability, and transparency in recommender systems [4, 5, 15, 38, 49, 50]. Our goal of minimizing negative experiences for the worst-case users is aligned with the fairness principle of not posing a disproportionate amount of risk to a sensitive group [21]. However, building such a framework for recommendations is not an obvious task without making some normative assumptions because users do not always act rationally [16]. That a user may not rationally optimize their own long-term interests, but rather sometimes act to maximize short-term reward is a well-studied principle in economics [16, 41].

Studying the effect of recommender systems on users and the platform is difficult for several reasons. First, a true audit of recommender systems often requires setting up real-world online experiments to carefully disentangle the causal effects of the recommender system policy from other exogenous variables that impact user choices [48]. Second, there are often multiple evaluation criteria that may have inherent trade-offs or disagreements [25]. While most of these metrics are based on measurements that are convenient for the system to make, the choice of a single metric is often debatable [27, 34].

One of the key methods to study and understand the dynamics and effects of recommender systems is the use of simulation studies. A few recent works have focused on using simulations to study the effect of recommender systems on the homogeneity of recommended items [8], the utility for the user [51], popularity of items [19, 62], and welfare of the item providers [39]. The underlying hypothesis is that if simulations are performed on a stylized user models that are close to accurate representations of reality, the phenomena may hold for real users as well. For our simulation, we use the Recsim framework [28] to develop user models and recommendation algorithms for reproducible experiments.

While the relevant literature emphasizes identifying the potential effects of recommender systems on its users and the platform, our work focuses on a novel idea of incorporating safety into learning algorithms for the RS to optimize for a positive experience, even in the worst-case.

3 FRAMEWORK: SAFE REINFORCEMENT LEARNING FOR RECOMMENDATIONS

In this section, we provide our problem setup for sequential recommendations as a Reinforcement Learning problem and define the healthiness of a recommendation policy using an exposure metric for the worst-case users.

3.1 Sequential Recommendation Framework

In our sequential recommendation framework, a user u arrives to the recommender system with an initial state $s_0(u)$. At any

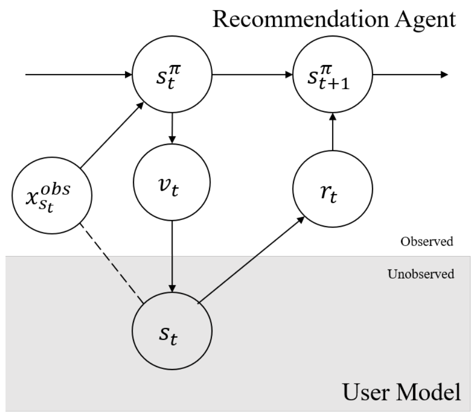

Figure 2: Sequential Recommendation Setup

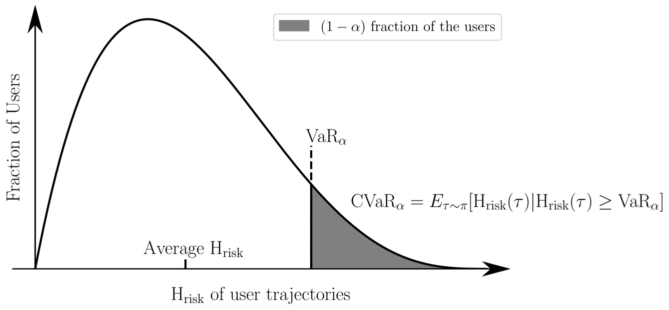

Figure 3: VaR_α is defined as the α^{th} percentile risk. CVaR_α is defined as the average of Health risks in the shaded region.

time t during the interaction with the recommender system, the system only has access to a subset of the user state, represented by $x^{\mathrm{obs}}(s_t(u))$, which it then uses to make a recommendation $v_{u,t}$ to the user. In this work, we will only consider recommending a single item $v_{u,t}$ from a set $\mathcal{D}$, however in general, $v_{u,t}$ could be a slate consisting of multiple items. The user responds to the recommendation with a feedback based on their current state, e.g. in the form of a click, rating etc., referred to as $r_{u,t}$, and also updates their internal state to $s_{t+1}(u)$ based on the recommended item and the feedback. Meanwhile the system uses $r_{u,t}$ to update its own internal state that we represent by s_t^π (see Figure 2). In summary, for each user u arriving to the system,

- At time step t in their trajectory, the user is at state $s_t(u) = \left(x^{\mathrm{obs}}(s_t(u)), x^{\mathrm{unobs}}(s_t(u))\right)$.
- The system recommends an item $v_{u,t} = \pi_t(x^{\mathrm{obs}}(s_t(u)))$.
- The user returns a feedback (e.g. click, rating, etc.) $r_{u,t} \leftarrow \mathrm{Reward}(s_t(u), v_{u,t})$.
- Meanwhile, the user model updates the state to $s_{t+1}(u)$ based on $s_t(u), v_{u,t}, r_{u,t}$ and its internal transition function.
- The recommender system updates its state to s_{t+1}^π.

This setup can be framed as a Markov Decision Process that can be solved using Reinforcement Learning (RL) where the primary goal is to find a policy π that maximizes the cumulative reward of the recommendations provided to users in the setup above i.e.

$$\pi^* = \underset{\pi}{\mathrm{argmax}}\, E_{u_i\sim\mathcal{U}}[R(\tau_i|\pi)]$$

where $\tau_i = (v_{i,t})_{t=1}^T = \left(v_{i,0}, v_{i,1}, \ldots, v_{i,T}\right)$ is the recommendation trajectory experienced by user u_i, and $R(\tau_i|\pi) = \frac{1}{T}\sum_{t=1}^T r_{i,t|\pi}$. In this work, we assume the length of each user trajectory to be fixed to T without the loss of generality, however, our setup can further be extended to the case where the user may leave the session before T steps. Since the cumulative reward $R(\tau_i)$ only aggregates the short-term rewards from the user u_i, in the next subsection, we describe the notion of healthiness of a user trajectory τ_i that focuses on a longer-term objective for the user, and later we will present an algorithm to optimize the RL policy (which we also refer to as a recommendation agent) to maximize the cumulative reward ($R(\tau_i)$) along with ensuring healthiness of the user trajectories.

3.2 Healthiness of User trajectories

As discussed in Section 1, our objective is to balance the positive feedback a user provides (e.g. click, ratings, etc.) with their exposure to potentially problematic content, which we will refer to as the *health risk*. In this section, we will formalize the notion of health risk for both average and worst-case user trajectories.

Health Risk of a recommendation trajectory. The unhealthiness of a user's experience can be quantified by assigning a health risk score to each item and aggregating the risk over the user's trajectory. Assume that each item v_j is associated with a health risk $h_{v_j} \in [0,1]$. This risk value may be assigned to the item by the user themselves at recommendation time, e.g,. a user trying to avoid a certain type of recommendation like high-sugar foods, or by an independent expert or a classifier, e.g. the amount of violence in a movie, etc. Now, we can define the health risk of a trajectory $\tau = (v_1, v_2, v_3, \ldots, v_T)$ to be $\mathrm{H}_{\mathrm{risk}}(\tau) = \frac{1}{T}\sum_{t=1}^T h_{v_t}$ i.e. the average health risk of items in the trajectory. In this work, we will assume that the health risk score is fixed for each item and can be aggregated over a user's trajectory as an average, however, in general, our method is agnostic to this design choice and can easily be extended to any arbitrary way of defining and aggregating risk over a user trajectory e.g. a nutritionist labeling each user trajectory with a personalized health risk score.

To define the health risk of a recommendation policy π using the health risk of individual user trajectories $\tau \sim \pi$, we could use a measure such as the average health risk over the user trajectories. However, for the goal we established in Section 1, we would like to define a notion of risk that measures the risk to the "worst-case" users.

Health Risk for the worst-case users. For a given percentile level α, the set of "worst-case" users is defined to be the set of users in the top $(1-\alpha)$ percentile of $\mathrm{H}_{\mathrm{risk}}$ values (Figure 3). For example, for $\alpha = 0.90$, we will define the set of users in the top 10 percentile by $\mathrm{H}_{\mathrm{risk}}$ as the worst-case users. To quantify the health risk that is borne by these users, we use the expected value of $\mathrm{H}_{\mathrm{risk}}$ for these users, which is referred to as *Conditional Value-at-Risk* at α (CVaR_α) [45], which can be formally defined as

$$\mathrm{CVaR}_\alpha(\mathrm{H}_{\mathrm{risk}}|\pi) \triangleq \mathbb{E}_{\tau\sim\pi}[\mathrm{H}_{\mathrm{risk}}(\tau)|\mathrm{H}_{\mathrm{risk}}(\tau) \geq VaR_\alpha(\mathrm{H}_{\mathrm{risk}}|\pi)] \tag{1}$$

where VaR_α refers to the Value-at-Risk and is equal to the risk value for the α^{th} percentile user i.e.

$$\text{VaR}_\alpha(\text{H}_{\text{risk}}|\pi) \triangleq \min h \text{ s.t. } P_\pi(\text{H}_{\text{risk}} \leq h) \geq \alpha$$

Figure 3 depicts a distribution of H_{risk} over user trajectories for an example policy π. The x-axis represents the health risk and the y-axis represents the frequency of users at each level of health risk. While VaR is the lower-bound value of the worst case users, CVaR is the average H_{risk} value of the users that lie above VaR. In this work, we choose CVaR as a measure of health risk over VaR because it CVaR is sensitive to the risk of the set of "worst-case" users while VaR is agnostic to the distribution of risk beyond itself [57]. Moreover, CVaR can tractably be optimized through a gradient descent method as we will see in Section 4.

Health Risk sensitive recommendation policies. Instead of solely maximizing the engagement-based reward for our sequential recommendation setup, we include a constraint into the learning problem that enforces a bound on the health risk (H_{risk}) of the "worst-case" users as defined by the CVaR metric. To this effect, the objective of our learning algorithm is constrained to a set of recommendation policies such that $\text{CVaR}_\alpha(\text{H}_{\text{risk}}|\pi)$ is bounded than some specified parameter δ.

$$\pi^*_{\alpha,\delta} = \underset{\pi}{\text{argmax}}\, \mathbb{E}_{u_i\sim\mathcal{U},\tau_i\sim\pi}[R(\tau_i|\pi)] \text{ s.t. } \text{CVaR}_\alpha(\text{H}_{\text{risk}}|\pi) \leq \delta$$

Using a Lagrange multiplier, this is equivalent to the following optimization objective

$$\pi^*_{\alpha,\delta} = \underset{\pi}{\text{argmax}}\, \min_{\lambda\geq 0} \mathbb{E}_{u_i\sim\mathcal{U},\tau_i\sim\pi}[R(\tau_i|\pi)] - \lambda\left(\text{CVaR}_\alpha(\text{H}_{\text{risk}}|\pi) - \delta\right)$$

In the following, we avoid minimization w.r.t. λ for a chosen risk level δ by choosing to steer the reward and risk trade-off by choosing a particular λ and then computing the corresponding δ afterwards. This means we solve the following

$$\pi^*_{\alpha,\lambda} = \underset{\pi}{\text{argmax}}\, \mathbb{E}_{u_i\sim\mathcal{U},\tau_i\sim\pi}[R(\tau_i|\pi)] - \lambda\, \text{CVaR}_\alpha(\text{H}_{\text{risk}}|\pi)) \quad (2)$$

and then recover the corresponding $\delta_{\alpha,\lambda} = \text{CVaR}_\alpha(\text{H}_{\text{risk}}|\pi^*_{\alpha,\lambda})$. Now that we have stated our goal as a joint objective, we will propose an algorithm to find the optimal policy $\pi^*_{\alpha,\lambda}$ in the next section.

4 METHOD: OPTIMIZING REWARD AND CONDITIONAL VALUE-AT-RISK (CVAR)

In this section, we introduce three different objectives to learn recommendation policies: optimizing purely for reward, balancing reward with average health of users, and balancing reward with the health of worst-case users (CVaR). We see how all three objectives can be optimized through adaptations of the REINFORCE algorithm [61], and later in Section 6, we will compare their performance trade-offs through simulation experiments.

Note that we still haven't described the architecture of our recommendation agent policy π. For our experimental evaluation, we will use a Recurrent Neural Network (RNN) based agent policy that we describe in Section 5.2, but for this section, it is enough to consider the class of policy functions that, at each time step, take a sequential user history as input to output a distribution over the feasible set of items, and are differentiable with respect to their parameters (denoted by θ). Now let's look at the different objectives to learn recommendation policies under such a policy class.

(1) **Reward Optimizing**: Our first objective takes the traditional approach of purely optimizing for reward, ignoring the health risk that users might experience. The objective of this policy can thus be described by:

$$\pi^{\theta^*} = \underset{\theta}{\text{argmax}} \quad \mathbb{E}_{u\sim\mathcal{U},\tau\sim\pi_\theta}[R(\tau)].$$

where $R(\tau)$ is the cumulative reward over the trajectory τ. To optimize the reward, we use the REINFORCE algorithm [61] which states that the gradient formula w.r.t. parameters θ using the likelihood ratio trick is

$$\nabla_\theta \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)] = \mathbb{E}_{u\sim\mathcal{U},\tau\sim\pi_\theta} \nabla_\theta \log P(\tau|\pi_\theta)\, [R(\tau)]\,. \quad (3)$$

In practice, when performing Empirical Risk Minimization over samples from the current policy, the expectation can be estimated as an average over a minibatch of user trajectories sampled using the policy π_θ and the gradient can be expressed as a sum over each time step in each trajectory as

$$\frac{1}{B} \sum_{\{\tau_1,\ldots,\tau_B\}\sim\pi_\theta} \left[\sum_{t=1}^{T} \nabla_\theta \log P(\tau_t|\pi_\theta) \left(\sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}\right)\right]$$

where $\gamma \in [0, 1]$ is a discount factor and B is the batch size. A more detailed derivation of this breakdown over multiple timesteps can be found in Schulman et al. [47].

(2) **Multiobjective with Average Health (Avg-Health-MO)**: In order to incorporate health similar to Equation 2, we first examine a multiobjective agent (see, e.g. [33, 36]) that balances the expected reward against the health risk the trajectory:

$$\pi^{\theta^*}_\lambda = \underset{\theta}{\text{argmax}} \quad \mathbb{E}_{u\sim\mathcal{U},\tau\sim\pi_\theta}[R(\tau)] - \lambda\mathbb{E}_{\tau\sim\pi_\theta}[\text{H}_{\text{risk}}(\tau)].$$

Here λ is a hyperparameter to control the agent's trade-off between the two objectives. Similar to the gradient formulation in case of optimizing reward only, we optimize this multiobjective reward by replacing $R(\tau)$ with $R(\tau) - \lambda\text{H}_{\text{risk}}(\tau)$ in Equation 3.

(3) **Multiobjective with Worst Case Health (CVaR-MO)**: Finally, we will examine an agent that optimizes for the health of worst-case users directly via the Conditional Value-at-Risk (CVaR). Our overall objective for a given α and λ (as in Equation 2) is

$$\pi^{\theta^*}_{\alpha,\lambda} = \underset{\theta}{\text{argmax}} \quad \mathbb{E}_{u\sim\mathcal{U},\tau\sim\pi_\theta}[R(\tau)] - \lambda\text{CVaR}_\alpha(\text{H}_{\text{risk}}|\pi) \quad (4)$$

However CVaR is a property of policy π and the definition of CVaR (Equation 1) cannot be decomposed as a sum over each trajectory τ_i. Hence, we follow Rockafellar et al. [45] to rewrite CVaR as:

$$\text{CVaR}_\alpha(\text{H}_{\text{risk}}|\pi) = \left[\nu_\alpha(\pi) + \frac{1}{1-\alpha}\mathbb{E}_\pi[(\text{H}_{\text{risk}}(\tau) - \nu_\alpha(\pi))^+]\right]$$

where $\nu_\alpha(\pi)$ is the VaR at α for the policy π, and $(x)^+ = \max(0, x)$. Since CVaR is thus expressed as an expectation over trajectories τ drawn from the policy π, we are able to combine the two terms of our objective such that:

$$\pi^{\theta^*}_{\alpha,\lambda} = \underset{\theta}{\text{argmax}}\ \mathbb{E}_{u\sim\mathcal{U},\tau\sim\pi_\theta}\Bigg[R(\tau) - \lambda\left(\nu_\alpha(\pi)\frac{1}{1-\alpha}(\mathrm{H}_{\text{risk}}(\tau) - \nu_\alpha(\pi))^+\right)\Bigg]$$

$$= \underset{\theta}{\text{argmax}}\ \mathbb{E}_{u\sim\mathcal{U},\tau\sim\pi_\theta} R'(\tau, \nu_\alpha(\pi))$$

$$(\text{where } R'(\tau,\nu_\alpha(\pi)) = R(\tau) - \lambda\left(\nu_\alpha(\pi) + \tfrac{1}{1-\alpha}\left(\mathrm{H}_{\text{risk}}(\tau) - \nu_\alpha(\pi)\right)^+\right))$$

Note that our modified reward function $R'(\tau, \nu_\alpha(\pi))$ is not only a function of trajectories τ but also the policy π because of the $\nu_\alpha(\pi)$ term and hence the gradient of the term with VaR is not as straightforward. One approach to write a policy gradient update that respects the dependence of ν on π is to treat it as a parameter of the model and then alternately update it with θ [11]. However, this has only been shown to converge well in smaller MDPs. In comparison, Tamar et al. [53, 54] introduce a much simpler approach that allows minibatching. They prove that as long as the risk score is continuous and bounded, one can approximate the gradient of CVaR for a minibatch by first estimating the VaR over the users in the minibatch and then plugging it into the modified reward function to compute a REINFORCE update for R' [61]. The approximation for the gradient estimate can hence be written as

$$\nabla_\theta \mathbb{E}_{u\sim\mathcal{U},\tau\sim\pi_\theta} R'(\tau, \nu_\alpha(\pi)) \approx \nabla_\theta \mathbb{E}_{u\sim\mathcal{U},\tau\sim\pi_\theta} R'(\tau, \tilde{\nu_\alpha})$$

where $\tilde{\nu_\alpha}$ is equal to the α-th percentile H_{risk} in a sample of trajectories from π_θ. The gradient estimate is only an approximation because $\tilde{\nu_\alpha}$ is a biased estimator of the $\mathrm{VaR}_\alpha(\pi)$ of the entire user population, however, the estimator $\tilde{\nu_\alpha}$ is a consistent estimator [54] i.e. choosing a large enough sample reduces the bias to 0. Hence, in our experiments, we choose a large enough minibatch (equal to 128) for each policy gradient step [54]. Now, using the log-likelihood ratio trick from Williams [61], the final formulation of the gradient step looks as follows

$$\nabla_\theta \mathbb{E}_{\tau\sim\pi_\theta} R'(\tau, \tilde{\nu_\alpha}) = \mathbb{E}_{\tau\sim\pi_\theta} \nabla_\theta \log P(\tau|\pi_\theta) \times \left[R(\tau) - \frac{\lambda}{1-\alpha}(\mathrm{H}_{\text{risk}}(\tau) - \tilde{\nu_\alpha})^+\right] \quad (5)$$

Similar to our previous objectives, during training, this expectation can be estimated as an average over sampled trajectories from π_θ to compute the empirical gradient estimate.

5 EXPERIMENTAL DESIGN

We conduct experiments on a simulation experiment framework built on the Movielens 1M (ML-1M) dataset. In this section, we first describe a simulation framework that allows us to study the disparate effect of a greedy recommendation policy (*Reward Optimizing*) on different user groups. Second, we define a Recurrent Neural Network (RNN) architecture for the recommendation agent that fits our framework.

5.1 Simulated Movie Recommendations

In the movie recommendation simulation, we consider the set of items to be the set of movies in the Movielens 1M dataset [22]. As a proxy for H_{risk}, we associate each movie with a Violence tag score from the Movielens Tag-genome Dataset [58], ranging from 0 to 1 for each movie. Our purpose in this investigation is not to suggest that violence is the right measure of health risk for movies (see e.g. [1, 13] for a more detailed discussion), but rather to illustrate how one might go about balancing risks for a given definition of health. We will assume that the health risk of a user trajectory is equal to the average of violence tag scores of the movies in the trajectory.

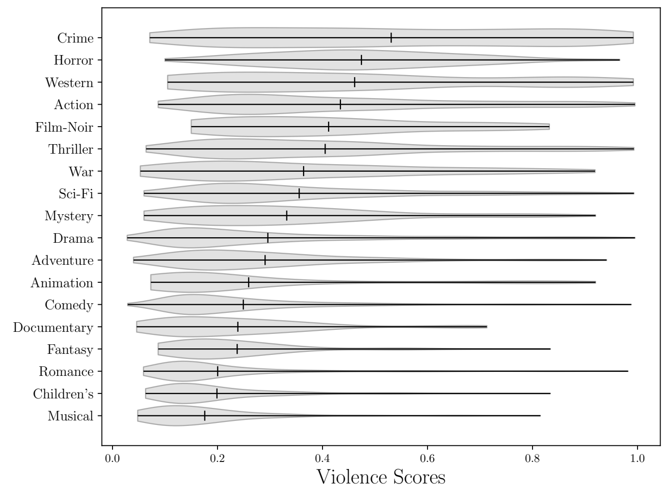

Figure 4: Distribution of Violence scores for different movie genres in ML-1M dataset. The genres are sorted in the decreasing order of the average violence score (H_{risk}) of their movies.

For the simulation experiments, each movie is also associated with a list of genres that we represent as a multi-hot vector embedding. The set of users are sampled from a set where each user is interested in exactly one movie genre, i.e. each user can be represented by a one-hot vector of size equal to the set of movie genres (say $\mathbf{u}_i$). To define the ground truth rating for each user and movie pair, we use two user models:

(A) **genre-only**: $r^*_{i,j} = \langle \mathbf{u}_i, \mathbf{v}_j \rangle$ i.e. the dot product between $\mathbf{u}_i$ is the one hot embedding of the user and $\mathbf{v}_j$ is the multi-hot embedding of the movie.
(B) **quality*genre**: $r^*_{i,j} = q_j \times \langle \mathbf{u}_i, \mathbf{v}_j \rangle$ where q_j is the average rating of movie j in ML-1M dataset normalized over all movies to be in [0,1].

Even though the sequential recommendation framework (Section 3) and the proposed methods (Section 4) apply to a more general setup, we make some simplifying assumptions here. First, the (unobserved) user state in the Movie Recommendation setup is stationary, i.e. a user's preference for a movie doesn't change based on their interactions with the agent. Secondly, the space of all possible user behaviors, even though combinatorial in size, is relatively small and thus not a significant test of generalization to new types of users at test time. We discover that even in this simplified environment, there are still substantial challenges for a recommender to balance health risk and user satisfaction. As such, we believe that these simplifications are valuable to more precisely study different approaches to address these challenges.

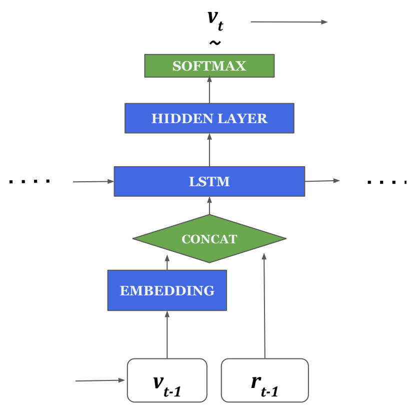

Figure 5: Architecture of a single step of the Recommendation Agent. At each time step for a user, the agent receives as input the previous recommendation and the corresponding reward, and outputs a distribution over items from which a recommendation is sampled.

In our simulation setup, each user starts with an empty history, i.e. their initial observed state is empty. Starting from $t = 0$, the agent recommends a movie at each time step and then adapts to the feedback from the user for the next step. The challenge for the agent, in our setup, is to figure out the preferred genre of the user, hopefully in the initial few steps of each trajectory, and then exploit that knowledge to collect high rewards for the remaining steps while not being allowed to recommend duplicate movies in a trajectory. A successful agent would be able to find a policy that can conduct this adaptive inference for each user at the test time.

Inherent Trade-offs. While this is a deliberately simple setup, we will see that it leads to non-trivial trade-offs between our two objectives. The essential tension in this simulation is that some genres have disproportionately more movies with violent content (e.g. Crime, War, etc.) as shown in Figure 4. The users that prefer these genres are likely to have trajectories with higher $\mathrm{H}_{\mathrm{risk}}$ and be in the top $(1 - \alpha)^{\mathrm{th}}$ percentile users in terms of risk. The CVaR-MO agent investigates whether we can tune the recommendations for these users to reduce their exposure to violent content with minimal impact on other users. In comparison, the average health agent (Avg-Health-MO) would suppress the movies with violence uniformly across all users, even though some trajectories may already have been sufficiently healthy.

5.2 Neural Recommendation Agent

For our experiments, we use a Recurrent Neural Network (RNN) based recommendation agent. RNNs are well-suited to the sequential decision-making tasks as they can model the temporal dynamics of user interaction histories and have thus been studied in several works on sequential recommendations [9, 26, 35, 55].

At each time step in the trajectory, our recommendation agent (see Figure 5) takes as input the previous recommendation and the corresponding reward. It maps the recommendation to a learned embedding of the movie, passes the embedding and the reward through an LSTM layer followed by a fully-connected layer with a softmax output. To prevent duplicate recommendations in a trajectory, a mask corresponding to the previously recommended movies is applied to the softmax before sampling the next recommendation. In our simulation setup, we consider users with no history (cold start), and hence the input at $t = 0$ of a user trajectory is a unique start token and a zero reward.

One key aspect of our setup is that when a user arrives at the recommender system, the agent has no information about the user. Hence, the agent's hidden state (the LSTM layer) has to implicitly infer the user preference through the user's actions starting from $t = 0$. Additionally, before training, the agent has no knowledge about the genres and qualities of the movies as well, and hence, it also needs to learn a meaningful representation for each movie in the embedding layer. This leads to an interesting cold-start sequential recommendation problem where even though we limit the user set to a few types of users, the agent has the challenge of learning both the movie and user representations from behavior patterns observed during the training phase.

Implementation Details: Our simulation framework is implemented using the Recsim framework [28], and agent models are defined using Tensorflow and Keras[1]. For the network architecture, we set the embedding size and hidden layer size to 20, while the output layer is of size 3706 (number of movies with genre information in our dataset). For training using the proposed methods in Section 4, each user interacts with the agent for $T = 20$ steps and their interaction trajectories are logged by the agent for a batch size of 128 (large minibatch to reduce the variance of the VaR estimates). We use an Adam optimizer [30] to perform a minibatch gradient descent update (Equations 3, 5) for 25000 gradient steps with a fixed learning rate of 0.0025 (selected through a grid search over learning rates in the range $[0.001, 0.1]$ for the *Reward Optimizing* method). For the reward, we use a discounting factor (Section 4) of $\gamma = 0.1$ for *genre-only* user model and $\gamma = 0$ for *quality*genre* user model (selected through a grid search over γ values in $\{0, 0.1, 0.2, \ldots, 0.9\}$ for the *Reward Optimizing* method), and we use the mean reward as the baseline term for all the policy gradient updates [60]. Throughout, when working with CVaR, we use $\alpha = 0.9$, i.e., we consider the top 10% users as those experiencing worst-case risk, and vary $\lambda \in [0, 100]$. For optimizing for Average $\mathrm{H}_{\mathrm{risk}}$, we vary $\lambda \in [0, 10]$.

5.3 Evaluation Setup

During the training phase, as we discussed earlier, the agent has to learn both the movie and user representations from user interactions. Once the agent is trained, we simulate the evaluation phase where the environment samples a user from the different types of users to simulate a trajectory of size T=20. At each step, the agent selects the recommendation that is equal to argmax of the Softmax layer (as compared to sampling during the training phase), and the environment returns the reward based on the identity of the user. In this work, we compare the following methods: (a) Reward Optimizing, (b) Avg-Health-MO, (c) CVaR-MO, (d) the filtering-based

[1]The experiments in this paper can be reproduced using the publicly available code at http://github.com/nnn/nnnn.nnnnnn

baseline agent (described below). Since training using reinforcement learning methods is often subject to high variance and instability [2, 56] and since the ability of the model to learn relies heavily on randomized exploration during training, we perform the training five times for each setting of λ in (b), (c) and (d), and drop two of the five models with the lowest combined training objective values (i.e. $R(\tau) - \lambda \text{CVaR}(\pi)$ for CVaR-MO, $R(\tau) - \lambda \text{H}_{\text{risk}}$ for Avg-Health-MO, and $R(\tau)$ for filtering-based baseline) to report the mean and standard error of metrics in the form of error bars during the evaluation.

Filtering Baseline. One intuitive way to impose a constraint on the health risk experienced by the users is to filter all movies above a certain threshold of violence score H_{risk}. We incorporate this approach as a baseline in our analysis as follows. During training and evaluation, we multiply a mask to the output layer so that the model will not recommend movies with $\text{H}_{\text{risk}} \geq \lambda$ (for a given $\lambda \in (0, 1]$). In our experiments, we vary the value of λ to train different Reward optimizing agents to obtain a trade-off between health and reward.

6 EMPIRICAL EVALUATION AND RESULTS

In this section, we present an empirical evaluation of the approaches described in Section 4 on the simulation environment for movie recommendations.

6.1 Reward-optimizing distributes health risks unequally

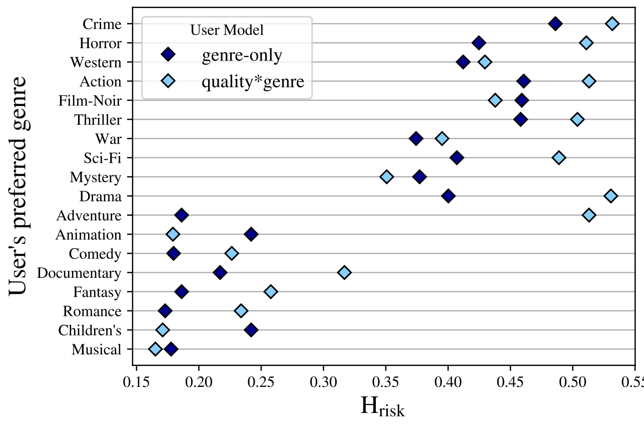

Figure 6: Distribution of the average H_{risk} for different types of users through the Reward Optimizing agent ($\lambda = 0$).

To examine how solely optimizing for reward leads to some users experiencing very unhealthy trajectories, we plot the distribution of average H_{risk} per user under the *Reward-optimizing* agent for the two user models (Section 5) in Figure 6. The distribution demonstrates that training without considering health at all leads to health risk (in this case, violence in movies) being unevenly distributed across users. Specifically, those users interested in genres with a higher prevalence of violent movies (see Figure 4) are exposed to a higher level of health risk, particularly when quality is incorporated into the user model. The quality*genre model leads to more users with $\text{H}_{\text{risk}} \geq 0.5$ which can be attributed to the fact that the genres corresponding to worst-case users have a higher correlation between quality and violence (0.30, 0.41, 0.27 respectively for Crime, Western, Action genres) than that of the genres on the left extreme of the H_{risk} axis (0.05, -0.02, 0.04 respectively for Musical, Children's, Romance genres). In the next sections, we investigate how our multiobjective formulations can be used to mitigate these risks to both the average and worst-case users.

6.2 Trading off between Health and Reward

To investigate the tradeoff between Health and Reward for the different approaches discussed in Section 4, we compare CVaR-MO (CVaR Multiobjective), Avg Health-MO (Average Health Multiobjective) and the filtering-based baseline agents for a range of λ values to control the agent's tradeoff between health and user ratings. Figure 7 visualizes the tradeoff in terms of both the average and worst-case health risk (CVaR). Starting from the performance of the *Reward optimizing* agent, i.e. $\lambda = 0$, as we increase the value of λ for each of the three methods, we see the average H_{risk} (and CVaR) decreases while also reducing the average ratings by the user. A better trade-off is one that allows H_{risk} to be reduced with a small reduction in Ratings. We observe that when evaluating health risk in terms of the worst case users (plots on the right), i.e. CVaR, the CVaR-MO agent leads to a significantly better rating vs. risk tradeoff than the Avg-Health-MO agent, as expected.

Surprisingly, even when evaluating in terms of average health, for most values of health risk, the CVaR-MO agent outperforms Avg-Health-MO (plots on the left). One possible explanation for this observation is the tension between the health risks for different genres. Some genres have disproportionately more movies with violent content (e.g. Crime, Action, etc. in Figure 4) and users who prefer these genres are more likely to be recommended more violent items. The Avg-Health-MO agent's objective function penalizes the recommendations of movies with violence uniformly across all users, even those with low prevalence of violence, and so it penalizes some high rating items for users interested in these low risk genres. In comparison, by definition, the CVaR-MO objective focuses more on users interested in genres with the highest prevalence of violence, and as a result, may need to make less of a trade-off between ratings and health for the rest of the users.

6.3 Variance of the H_{risk} distribution

To further investigate the differences between these two training objectives, we compare two models trained on the genre-only user environment setup–one trained using CVaR-MO ($\lambda = 10.0$) and Avg-Health-MO ($\lambda = 0.6$), both of which achieve a similar average H_{risk} (0.122 and 0.120 respectively) during evaluation. In Figure 8, we visualize the risk score distributions across different users for these two sample models. We observe that the CVaR-MO agent is able reduce the variance of the H_{risk}, despite having a similar mean to the Avg-Health-MO agent. While the CVaR-MO agent achieves this by optimizing for the worst case users, the objective used by the Avg-Health-MO agent only brings the average down without being sensitive to the worst cases.

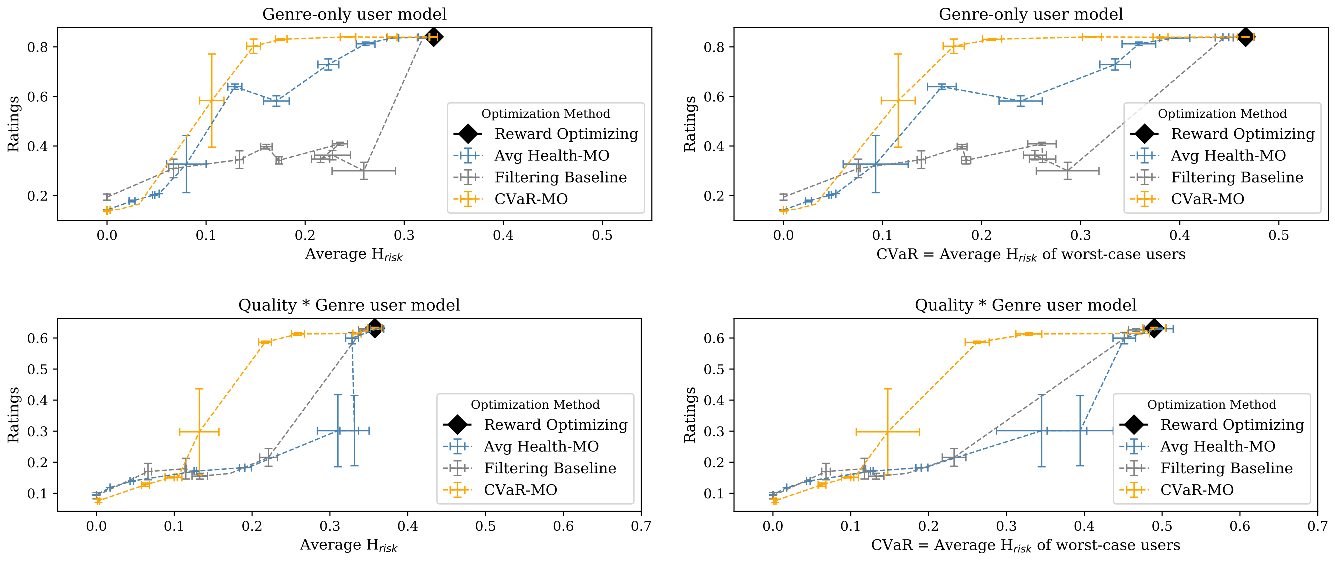

Figure 7: Trade-off between health risk and average ratings for recommendation policies trained on genre-only and quality*genre user models. A model that performs better is one more in the top-left corner of each plot, i.e. if it has higher ratings and lower health risk.

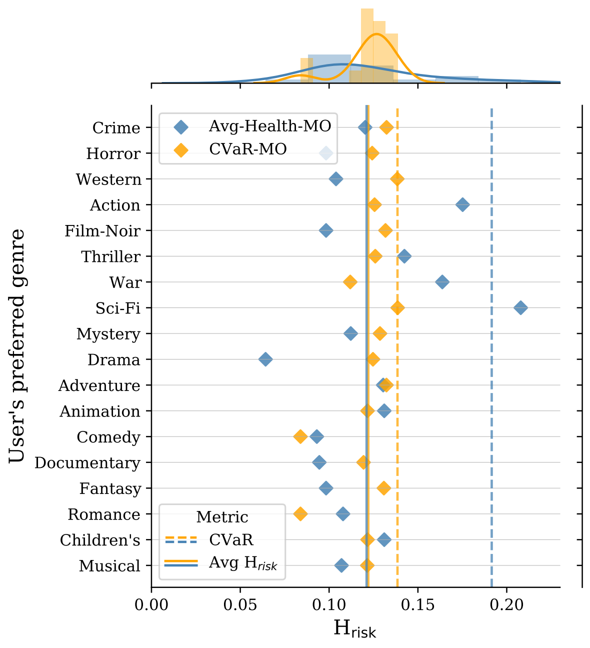

Figure 8: Comparison between the distribution of health risks over users for two agent models that achieve similar average H_{risk} (solid line) but significantly different $CVaR_{0.9}(H_{risk})$ (dotted line).

7 CONCLUSION

In this paper, we have proposed a new goal for designing responsible recommender systems to improve the healthiness of *all* users' experiences. We explore a range of approaches for recommendation systems that seek to balance two competing objectives: optimizing for positive feedback and limiting cumulative exposure to unhealthy items. In particular, we propose taking a safe reinforcement learning approach that doesn't just balance the amount healthiness of the user experience on average but also in the worst case. We propose a Movielens based simulation environment to study this trade-off and demonstrate that a purely rating optimizing agent could lead to unhealthy outcomes disproportionately shared by a subset of users. We also show that an agent who optimizes for both ratings and average health can, indeed, improve the health outcomes for some users but fails to account for those users who are most likely to experience unhealthy content. For this, we turn to an agent that balances ratings against tail risk, showing that it not only improves the health of worst-case users but also leads to better tradeoffs in the average case as well. We believe this is an important step in responsibly designing recommender systems that can benefit *all* users and hope it can provide a foundation for future research to improve more aspects of the user experience.

REFERENCES

[1] Craig A Anderson, Leonard Berkowitz, Edward Donnerstein, L Rowell Huesmann, James D Johnson, Daniel Linz, Neil M Malamuth, and Ellen Wartella. 2003. The influence of media violence on youth. *Psychological science in the public interest* 4, 3 (2003), 81–110.
[2] Oron Anschel, Nir Baram, and Nahum Shimkin. 2017. Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning *(Proceedings of Machine Learning Research)*, Doina Precup and Yee Whye Teh (Eds.), Vol. 70. PMLR, International Convention Centre, Sydney, Australia, 176–185. http://proceedings.mlr.press/v70/anschel17a.html
[3] Saeideh Bakhshi, Partha Kanuparthy, and Eric Gilbert. 2014. Demographics, weather and online reviews: A study of restaurant recommendations. In *Proceedings of the 23rd international conference on World wide web*. 443–454.
[4] Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Li Wei, Yi Wu, Lukasz Heldt, Zhe Zhao, Lichan Hong, Ed H Chi, et al. 2019. Fairness in recommendation ranking through pairwise comparisons. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining*. 2212–2220.

[5] Asia J Biega, Krishna P Gummadi, and Gerhard Weikum. 2018. Equity of attention: Amortizing individual fairness in rankings. In *The 41st international acm sigir conference on research &amp; development in information retrieval*. 405–414.

[6] Vivek S Borkar. 2001. A sensitivity formula for risk-sensitive cost and the actor–critic algorithm. *Systems & Control Letters* 44, 5 (2001), 339–346.

[7] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, Yunfei Bai, Matthew Kelcey, Mrinal Kalakrishnan, Laura Downs, Julian Ibarz, Peter Pastor, Kurt Konolige, et al. 2018. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 4243–4250.

[8] Allison JB Chaney, Brandon M Stewart, and Barbara E Engelhardt. 2018. How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 224–232.

[9] Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H. Chi. 2019. Top-K Off-Policy Correction for a REINFORCE Recommender System. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* (Melbourne VIC, Australia) *(WSDM '19)*. Association for Computing Machinery, New York, NY, USA, 456–464. https://doi.org/10.1145/3289600.3290999

[10] Xinshi Chen, Shuang Li, Hui Li, Shaohua Jiang, Yuan Qi, and Le Song. 2018. Neural Model-Based Reinforcement Learning for Recommendation. *CoRR* abs/1812.10613 (2018). arXiv:1812.10613 http://arxiv.org/abs/1812.10613

[11] Yinlam Chow and Mohammad Ghavamzadeh. 2014. Algorithms for CVaR Optimization in MDPs. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3509–3517. http://papers.nips.cc/paper/5246-algorithms-for-cvar-optimization-in-mdps.pdf

[12] Stefano P Coraluppi and Steven I Marcus. 1999. Risk-sensitive and minimax control of discrete-time, finite-state Markov decision processes. *Automatica* 35, 2 (1999), 301–309.

[13] Gordon Dahl and Stefano DellaVigna. 2009. Does movie violence increase violent crime? *The Quarterly Journal of Economics* 124, 2 (2009), 677–734.

[14] Gabriel Dulac-Arnold, Richard Evans, Hado van Hasselt, Peter Sunehag, Timothy Lillicrap, Jonathan Hunt, Timothy Mann, Theophane Weber, Thomas Degris, and Ben Coppin. 2015. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679* (2015).

[15] Michael D Ekstrand, Robin Burke, and Fernando Diaz. 2019. Fairness and discrimination in retrieval and recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1403–1404.

[16] Drew Fudenberg and David K Levine. 2006. A dual-self model of impulse control. *American economic review* 96, 5 (2006), 1449–1476.

[17] Javier García, Fern, and o Fernández. 2015. A Comprehensive Survey on Safe Reinforcement Learning. *Journal of Machine Learning Research* 16, 42 (2015), 1437–1480. http://jmlr.org/papers/v16/garcia15a.html

[18] Peter Geibel and Fritz Wysotzki. 2005. Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Artificial Intelligence Research* 24 (2005), 81–108.

[19] Fabrizio Germano, Vicenç Gómez, and Gaël Le Mens. 2019. The few-get-richer: a surprising consequence of popularity-based rankings?. In *The World Wide Web Conference*. 2764–2770.

[20] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM* 35, 12 (1992), 61–70.

[21] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*. 3315–3323.

[22] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[23] Chen He, Denis Parra, and Katrien Verbert. 2016. Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities. *Expert Systems with Applications* 56 (2016), 9–27.

[24] Matthias Heger. 1994. Consideration of risk in reinforcement learning. In *Machine Learning Proceedings 1994*. Elsevier, 105–111.

[25] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)* 22, 1 (2004), 5–53.

[26] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[27] Liangjie Hong and Mounia Lalmas. 2019. Tutorial on Online User Engagement: Metrics and Optimization. In *Companion Proceedings of The 2019 World Wide Web Conference*. 1303–1305.

[28] Eugene Ie, Chih-wei Hsu, Martin Mladenov, Vihan Jain, Sanmit Narvekar, Jing Wang, Rui Wu, and Craig Boutilier. 2019. RecSim: A Configurable Simulation Platform for Recommender Systems. *arXiv preprint arXiv:1909.04847* (2019).

[29] Eugene Ie, Vihan Jain, Jing Wang, Sanmit Narvekar, Ritesh Agarwal, Rui Wu, Heng-Tze Cheng, Tushar Chandra, and Craig Boutilier. 2019. SlateQ: A tractable decomposition for reinforcement learning with recommendation sets. (2019).

[30] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[31] Joseph A Konstan and John Riedl. 2012. Recommender systems: from algorithms to user experience. *User modeling and user-adapted interaction* 22, 1-2 (2012), 101–123.

[32] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 426–434.

[33] Shie Mannor and Nahum Shimkin. 2002. The steering approach for multi-criteria reinforcement learning. In *Advances in Neural Information Processing Systems*. 1563–1570.

[34] Rishabh Mehrotra, Ashton Anderson, Fernando Diaz, Amit Sharma, Hanna Wallach, and Emine Yilmaz. 2017. Auditing search engines for differential satisfaction across demographics. In *Proceedings of the 26th international conference on World Wide Web companion*. 626–633.

[35] R Mehrotra, AH Awadallah, M Shokouhi, E Yilmaz, I Zitouni, A El Kholy, and M Khabsa. 2017. Deep Sequential Models for Task Satisfaction Prediction. In *CIKM'17: PROCEEDINGS OF THE 2017 ACM CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT*. ACM, 737–746.

[36] Rishabh Mehrotra, Niannan Xue, and Mounia Lalmas. 2020. Bandit based Optimization of Multiple Objectives on a Music Streaming Platform. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining*. 3224–3233.

[37] Oliver Mihatsch and Ralph Neuneier. 2002. Risk-sensitive reinforcement learning. *Machine learning* 49, 2-3 (2002), 267–290.

[38] Silvia Milano, Mariarosaria Taddeo, and Luciano Floridi. 2020. Recommender systems and their ethical challenges. *AI & SOCIETY* (2020), 1–11.

[39] Martin Mladenov, Elliot Creager, Omer Ben-Porat, Kevin Swersky, Richard Zemel, and Craig Boutilier. 2020. Optimizing Long-term Social Welfare in Recommender Systems: A Constrained Matching Approach. (2020).

[40] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[41] Alexander Peysakhovich. 2019. Reinforcement learning and inverse reinforcement learning with system 1 and system 2. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*. 409–415.

[42] John Rawls. 2001. *Justice as fairness: A restatement*. Harvard University Press.

[43] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. 175–186.

[44] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. Introduction to recommender systems handbook. In *Recommender systems handbook*. Springer, 1–35.

[45] R Tyrrell Rockafellar, Stanislav Uryasev, et al. 2000. Optimization of conditional value-at-risk. *Journal of risk* 2 (2000), 21–42.

[46] Makoto Sato, Hajime Kimura, and Shibenobu Kobayashi. 2001. TD algorithm for the variance of return and mean-variance reinforcement learning. *Transactions of the Japanese Society for Artificial Intelligence* 16, 3 (2001), 353–362.

[47] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-Dimensional Continuous Control Using Generalized Advantage Estimation. arXiv:1506.02438 [cs.LG]

[48] Amit Sharma, Jake M Hofman, and Duncan J Watts. 2015. Estimating the causal impact of recommendation systems from observational data. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*. 453–470.

[49] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of exposure in rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining*. 2219–2228.

[50] Ashudeep Singh and Thorsten Joachims. 2019. Policy learning for fairness in ranking. In *Advances in Neural Information Processing Systems*. 5426–5436.

[51] Wenlong Sun, Olfa Nasraoui, and Patrick Shafto. 2018. Iterated algorithmic bias in the interactive machine learning process of information filtering. In *10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K 2018*. SciTePress, 110–118.

[52] Richard S Sutton, Andrew G Barto, and Ronald J Williams. 1992. Reinforcement learning is direct adaptive optimal control. *IEEE Control Systems Magazine* 12, 2 (1992), 19–22.

[53] Aviv Tamar, Yinlam Chow, Mohammad Ghavamzadeh, and Shie Mannor. 2015. Policy Gradient for Coherent Risk Measures. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 1468–1476. http://papers.nips.cc/paper/5923-policy-gradient-for-coherent-risk-measures.pdf

[54] Aviv Tamar, Yonatan Glassner, and Shie Mannor. 2015. Optimizing the CVaR via sampling. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

[55] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. 17–22.

[56] Sebastian Thrun and Anton Schwartz. 1993. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 Connectionist Models Summer School Hillsdale, NJ. Lawrence Erlbaum*.

[57] Stan Uryasev. [n.d.]. VaR vs CVaR in risk management and optimization.

[58] Jesse Vig, Shilad Sen, and John Riedl. 2012. The tag genome: Encoding community knowledge to support novel interaction. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 2, 3 (2012), 1–44.

[59] Xinxi Wang, Yi Wang, David Hsu, and Ye Wang. 2014. Exploration in interactive personalized music recommendation: a reinforcement learning approach. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 11, 1 (2014), 1–22.

[60] Lex Weaver and Nigel Tao. 2001. The optimal reward baseline for gradient-based reinforcement learning. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*. 538–545.

[61] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

[62] Sirui Yao, Yoni Halpern, Nithum Thain, Xuezhi Wang, Kang Lee, Flavien Prost, Ed H. Chi, Jilin Chen, and Alex Beutel. 2020. Measuring Recommender System Effects with Simulated Users. *FATES at WWW* (2020).

[63] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)* 52, 1 (2019), 1–38.

[64] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*. 167–176.