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

ASHUDEEP SINGH*, Cornell University
YONI HALPERN, NITHUM THAIN, KONSTANTINA CHRISTAKOPOULOU, ED H. CHI, JILIN CHEN, ALEX BEUTEL, Google Research

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 leading us to examine a distributional notion of recommendation safety. Thus, 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 novel research simulation environment motivated by a movie recommendation setting that considers exposure to violence as a proxy for unhealthy recommendations. We demonstrate how 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, but it is also valuable in many applications for them to create positive and healthy user experiences. For example, we may want repeated recommendations for restaurants to both be enjoyable to users and also mostly healthy; recommending only ice cream, while enjoyable, is likely not a good experience. 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].

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 violence are not binary but rather continuous-valued quantities and in designing a recommender, a positive user experience may include the occasional interaction with these types of items, but repeated exposure may ultimately lead to a negative user experience. Hence, in formulating the problem, we examine two competing objectives: optimizing for positive feedback, like positive ratings, and also for limiting long-term exposure to unhealthy items.

We simulate a simplified form of the problem using the Movielens dataset [19] with users who prefer exactly one genre of movie, and we consider violence as a problematic feature that a recommender may want to limit exposure to in the long-term. In our analysis, we discover that jointly optimizing for avoiding violence and improving user ratings can help improve the average case, but fails to account for a small minority of users who indeed prefer genres that are highly correlated with violence, like crime or war films. We then describe a modified objective that can optimize for a distributional view of exposure to violence, achieving a more positive experience even for users whose tastes are strongly correlated with this violence.

Contributions: 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. We show how to optimize this metric in a sequential recommendation setting with a policy gradient algorithm (Section 4). We formulate a simplified

*Work conducted while the author was an intern at Google Research. Correspondence address: ashudeep@cs.cornell.edu

simulation environment, using the movielens dataset [19] that exposes a tension between user rating feedback and exposure to violence. Finally, we evaluate the learning algorithm in this simulation environment to examine aspects of its behavior and performance (Section 5).

2 RELATED WORK

Conventionally, recommender systems (RS) have used collaborative filtering to build an accurate model of short-term feedback from a user by using their past history [17, 31]. For this task, latent factor models and matrix factorization techniques have shown promise in many real-world rating prediction tasks [26]. Recently there has been a growing interest in studying interactive recommender systems that treat recommendations as a sequence of interactions with users [20]. Given the considerable success of Reinforcement Learning (RL) in games [30], robotics [5], and physical system control [39], it has become a common framework to train recommenders that optimize user feedback over the entire sequence [8, 44, 48]. 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 [7]. Another challenge is the size of the action space that can range from hundreds to millions depending on the recommendation task [12, 25].

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 it is crucial that the agent avoids physical damage while completing tasks [28]. 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 [10, 21], variance of the rewards [33], worst-case outcome [4], probability of an agent producing a high-risk trajectory [28] or reaching an unsafe state [15] (see García et al. [14] for a comprehensive survey). In this work, we specifically use the percentile risk criteria defined on a measure we will refer to as the health risk of the recommendation trajectories (Section 3) and adapt the Policy Gradient algorithm from Tamar et al. [41] to train recommendation agents (Section 4).

Our work is motivated by the growing community of research in fairness, accountability, and transparency in recommender systems [2, 3, 13, 36, 37]. 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 [18]. Studying the effect of recommender systems on users and the platform is difficult for a number of reasons. First, a true audit of recommender systems often requires setting up real-world online experiments to carefully disentangle the effects of the recommender system policy from other exogenous variables that impact user choices [35]. Second, there are often multiple evaluation criteria that may have inherent trade-offs. While most of these metrics are based on measurements that are convenient for the system to make, the choice of any one metric is often debatable [23]. 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 [6], the utility for the user [38], popularity of items [16, 47], and welfare of the item providers [29]. The hypothesis is that if stylized user models built for simulations are close to reality, the simulations will demonstrate the same phenomena that occur with real users. For our simulation, we use the Recsim framework [24] to develop user models and recommendation algorithms for reproducible experiments. While the related works emphasize identifying the potential effects of recommender systems on its user and the platform, our research focuses on 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 worst-case users.

3.1 Background: Recommendation as Reinforcement Learning

In our setup, for each user session i, a user $u_i \in \mathcal{U}$ arrives to the system and seeks recommendations from a set of items $\mathcal{D}$ (e.g. movies). At each time step $t \in \{1 \dots T\}$ of their trajectory, the recommendation policy, denoted by π, recommends an item $v_{i,t}$ from $\mathcal{D}$ based on their interactions in the trajectory so far. For each recommendation $v_{i,t}$, the user provides feedback to the recommender system (e.g. click or rating) based on their current preference for the item. We will refer to this feedback as the reward and denote it by $r_{i,t}$. In short, the user u_i experiences a trajectory $\tau_i = (v_{i,1}, v_{i,2}, \dots, v_{i,T})$ while returning reward feedback $(r_{i,1}, r_{i,2}, \dots, r_{i,T})$. The primary goal of the recommendation system is to maximize the cumulative reward of the recommendations provided to users i.e. $\frac{1}{N}\sum_{i=1}^{N} R(\tau_i)$ where $R(\tau_i) = \frac{1}{T}\sum_{t=1}^{T} r_{i,t}$ and N is the number of users. Note that, in general, one could extend this formulation to a set of recommendations, e.g. as a slate or a ranking, at each time step t.

The goal of building an effective recommender system for this sequential recommendation setup is to learn a recommendation policy π that maps a user's current trajectory (observed state) to a recommendation (action) while maximizing the cumulative reward over a distribution of users. In this work, we assume the length of each user trajectory to be fixed to T without the loss of generality. Our setup can further be extended to the case where the user may leave the session before T steps.

3.2 Problem Formulation: Health in Recommendations

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

3.2.1 Health Risk of a recommendation trajectory. The unhealthiness of each 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 $h_{v_j} \in [0, 1]$, e.g. the amount of violence in a movie, some measure of the unhealthiness of a food option, etc. We then define the health risk of a trajectory $\tau = (v_1, v_2, v_3, \dots, v_T)$ to be $\mathrm{H}_{\mathrm{risk}}(\tau) = \frac{1}{T}\sum_{t=1}^{T} h_{v_t}$, the average health risk of items in this trajectory. In this work, we assume that 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 the choice of the health risk metric for recommendations and can easily be extended to other ways of aggregating over a trajectory.

3.2.2 Health Risk for the worst-case users. While optimizing to minimize health risk may be beneficial, it doesn't necessarily capture each user's experience and hence we extend the definition of health risk specifically to worst-case users through a distributional criteria. For a given percentile level α, the worst-case users are defined to be the set of users in the top $(1-\alpha)$ percentile of $\mathrm{H}_{\mathrm{risk}}$ values (Figure 1). To quantify the health risk that is borne by these users, we use the expected value of $\mathrm{H}_{\mathrm{risk}}$ for these users with the highest $\mathrm{H}_{\mathrm{risk}}$, which is referred to as Conditional Value-at-Risk at α (CVaR_α), which can be defined as

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

where VaR_α refers to the Value-at-Risk and is equal to the cut-off value for the top-α percentile of users i.e. $VaR_\alpha(\text{H}_{\text{risk}}|\pi) \triangleq \min h$ s.t. $P_\pi(\text{H}_{\text{risk}} \leq h) \geq \alpha$. Figure 1 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 of H_{risk} values of the users that lie above VaR .

4 METHOD: OPTIMIZING REWARD AND CVAR

In this section, we introduce policies defined by three different objectives: optimizing purely for reward, balancing reward with average health, and balancing reward with the health of worst-case users (CVaR). We see how all three policies can be learned through adaptations of the REINFORCE algorithm and in Section 6, we compare their performance tradeoffs in simulation.

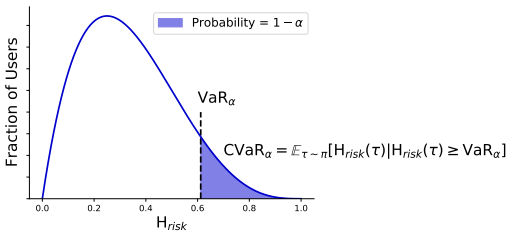

Fig. 1. VaR_α is defined as the α^{th} percentile risk. CVaR_α is defined as the average of Health risks in the shaded region.

(1) **Reward Optimizing**: Our first policy 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:

$$\pi_{\theta^*} = \underset{\theta}{\text{argmax}} \quad \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)]$$

where $R(\tau)$ is the cumulative reward over the trajectory τ. To optimize the reward, we train the REINFORCE algorithm [45] which states that the gradient formula using the likelihood ratio trick is

$$\nabla_\theta \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)] = \mathbb{E}_{\tau\sim\pi_\theta} \nabla_\theta \log P(\tau|\pi_\theta)\,[R(\tau)]$$

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 trajectories sampled from 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 γ is a discount factor $\in (0, 1]$ and B is the batch size. A more detailed derivation can be found in Schulman et al. [34].

(2) **Multiobjective with Average Health**: In order to incorporate health, we first examine a multiobjective agent (see, e.g. [27]) that balances the expected reward against the average health across the trajectory:

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

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 $R(\tau) - \lambda\text{H}_{\text{risk}}(\tau)$.

(3) **Multiobjective with Worst Case Health**: Finally, we will examine an agent that optimizes for the health of worst-case users directly via the Conditional Value-at-Risk (CVaR). The overall objective of our recommender system is thus:

$$\pi_{\theta^*} = \underset{\theta}{\text{argmax}} \quad \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)] - \lambda\text{CVaR}_{\,\alpha}(\text{H}_{\text{risk}}|\pi) \tag{2}$$

Since CVaR is a property of policy π rather than each trajectory τ, we follow Rockafellar et al. [32] to rewrite it 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:

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

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 [9]. However, this has only been shown to converge well in smaller MDPs. In comparison, Tamar et al. [40, 41] introduce a much simpler approach that allows minibatching. The work proves that as long as the risk score is continuous and bounded, one can approximate the gradient of CVaR for a minibatch gradient descent by first estimating the VaR over the minibatch, and then plug it into the modified reward function to compute a REINFORCE update for R' [45]. The approximation for the gradient estimate can hence be written as

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

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 consistent i.e. choosing a large enough minibatch reduces the bias to 0. Hence, in our experiments, we choose a large enough minibatch (equal to 128) for each policy gradient step [41]. Now, using the log-likelihood ratio trick from Williams [45], the final formulation of the gradient step looks as follows

$$\nabla_\theta \mathbb{E}_{\tau\sim\pi_\theta} R'(\tau, \tilde{\nu_\alpha}) = \mathbb{E}_{\tau\sim\pi_\theta} \nabla_\theta \log P(\tau|\pi_\theta)\left[R(\tau) - \frac{\lambda}{1-\alpha}(\mathrm{H}_{\text{risk}}(\tau) - \tilde{\nu_\alpha})^+\right] \tag{3}$$

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

To evaluate our proposed approach for maximizing the health for the worst-case users, we present a set of experiments on a simulated movie recommendation setup.

5.1 Simulated Movie Recommendation Setup

We consider a set of simulated users on a subset of movies in the Movielens 1M dataset [19]. We associate each movie with a Violence tag score from the Movielens Tag-genome Dataset [43], ranging from 0 to 1 for each movie, which we choose as a proxy for H_{risk}. Our purpose in this investigation is not to suggest that violence is the right measure of health risk for movies (see e.g. [1, 11] for a more detailed discussion), but rather to illustrate how one might go about balancing these risks for a given definition of health. For the simulation experiments, we select a 10% subset (around 400 movies) such that the distribution of violence scores for the subset is uniform between 0 and 1. Each movie is also associated with a list of genres that we represent as a multi-hot vector embedding for the movie.

We simulate a set of users such that each user is interested in exactly one movie genre, which is represented by a one-hot vector of size equal to the set of movie genres. The rating of a user for a particular movie is thus expressed as the dot product between the user and the movie vector.

In our simulation setup, each user starts with an empty history and the agent adapts to the feedback as the user interacts with the recommendation at each time-step. The challenge for the agent 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 where the agent is not permitted to recommend duplicate movies in a trajectory.

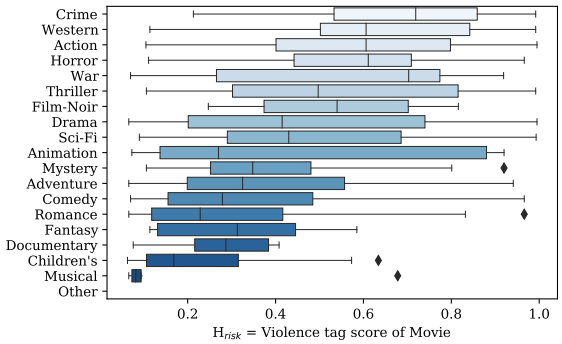

Fig. 2. Distribution of Violence scores for different movie genres in our corpus.

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 2. The users that prefer these genres are likely to have trajectories with higher H_{risk} and be in the top $1-\alpha$th percentile users in terms of risk. Our worst-case 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 would suppress the movies with violence uniformly across all users, even though some trajectories may already have been sufficiently healthy.

5.2 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 [7, 22, 42, 46].

At each time step in the trajectory, our policy network takes as input the previous recommendation and the corresponding reward. It maps the recommendation to a learned embedding of the movie, passes it 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. The input at $t = 0$ of a user trajectory is a dummy start token and a zero reward. Figure 3 shows the structure of our neural network agent.

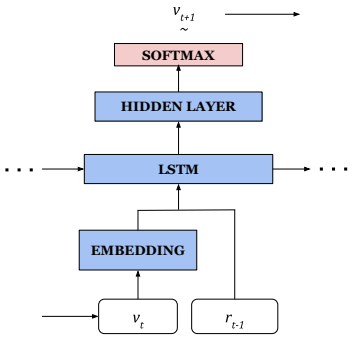

Fig. 3. Architecture of a single step of the Recommendation Agent. At each time step for a user, the agent receives as input the previous recommendation and the corresponding reward.

Note that when a user arrives at the recommender system, the agent has no information about the user. The recurrent hidden state implicitly learns the user preference through the history of user actions. Additionally, before training, the agent has no information about the movies (and their genres). It also needs to learn a meaningful representation for each movie in the embedding layer. Even though our user set is limited in size, the agent has the challenge of learning both the movie representations and user behavior during training.

Implementation Details: We run our training algorithm for the movie recommendation setup above, where each user interacts with the agent for $T = 20$ steps. For the network architecture, we set the embedding size to 32, and the hidden layer size to 128, and use an Adam optimizer for the gradient in Equation 3 with a line search on learning rates in $\{0.001, 0.0005, 0.00025\}$. To optimize for CVaR, we use $\alpha = 0.9$ and use λ values $\in [0, 100]$. In the results section, we refer to the two multiobjective agents with the CVaR and the average health objectives as *CVaR-MO* and *Avg-Health-MO*

respectively and denote the agent trained with $\lambda = 0$ as the *Reward optimizing agent*. We implement our simulation using the Recsim framework [24] and used Tensorflow and Keras to define and train our neural network models[1].

6 RESULTS

In this section, we present our empirical evaluation on the simulation environment for movie recommendations.

6.1 How is the health risk distributed across users?

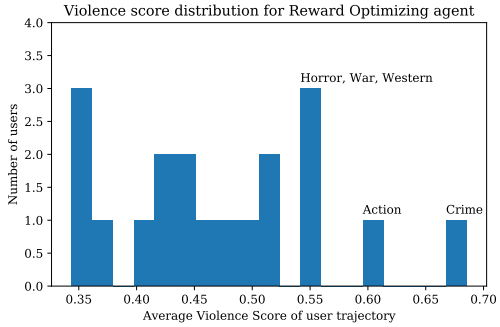

Fig. 4. Distribution of H_{risk} for different users with Reward Optimizing agent ($\lambda = 0$). The annotations indicates user genres of the users with the highest average violence scores.

Figure 4 shows the distribution of health risk induced by a *Reward optimizing agent* across our users. The distribution also demonstrates that when the recommender system is only focused on user ratings, the amount of health risk (in this case, violence in movies) is very unevenly distributed across users. Moreover, those users who are interested in genres with a high level of violent movies (see Figure 2) are exposed to a higher level of such health risk. 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 What is the trade-off between Health and Reward?

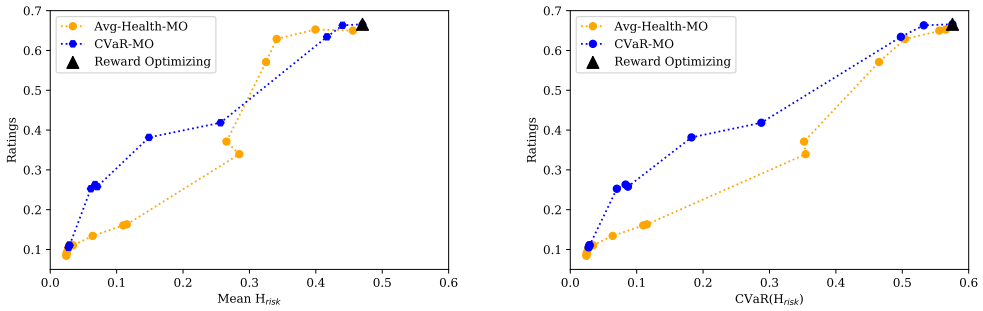

Fig. 5. Left: Average H_{risk} vs Ratings tradeoff comparison between CVaR-MO and Avg-Health-MO agents for different values of λ. Right: $CVaR_{\alpha=0.9}(H_{risk})$ vs Ratings trade-off for the two methods.

To investigate the tradeoff between Health and Reward for the CVaR-multiobjective (CVaR-MO) and Average Health multiobjective (Avg-Health-MO) approaches in Section 3, we trained recommendation agents for different values of λ. Figure 5 illustrates the tradeoff in terms of both mean and tail health risk. We observe that, as expected, when evaluating health risk purely in terms of the tail metric (CVaR), the CVaR-MO agent leads to a better rating vs. risk tradeoff than the Avg-Health-MO agent. Surprisingly, when evaluating in terms of average health, there are regions where the CVaR-MO agent outperforms as well, namely those with high λ and correspondingly lower health risk. 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, War, etc.) and thus the users that prefer these genres are more likely to be recommended more violent items. The CVaR-MO agents only include these users in the

[1] https://github.com/nnnnn/nnnnn–We plan to update the URL with a public repository of our code in the final version of the paper.

optimization objective and thus in principle should only effect the recommendations for these users with minimal impact on the recommendations of other users. By contrast, the Avg-Health-MO agents decrease the recommendations of movies with violence uniformly across all users, even though some trajectories may already have been sufficiently healthy and thus, during the exploration phase in training, it loses out on some high rating items for the users that weren't at significant risk.

6.3 How does the trade-off compare for different types of users?

In Section 6.1, we identified that users who prefer the *Crime* genre encounter the highest amount of violent movies in the case of a purely *Reward Optimizing agent*. To deepen our understanding of the differences between our agent designs in terms of the chosen objective, in Figure 6 we compare the trade-offs induced by these agents across a range of λ values for a Crime genre user with another user who prefers the Drama genre, which has a much lower average violence score (see Figure 2). Notice that, in terms of rating, the difference between these models is far more significant for the Drama user than for Crime. In particular, for intermediate lambda values, the Avg-Health-MO agent recommends movies with a substantially lower rating for Drama users. As the agents explore the space of movies during training, the Avg-Health-MO agent gains sub-optimal reward by simply recommending healthier (less violent) movies rather than exploring the higher rating alternatives, while the CVaR-MO agent only rewards such health improvements for the worst-case users.

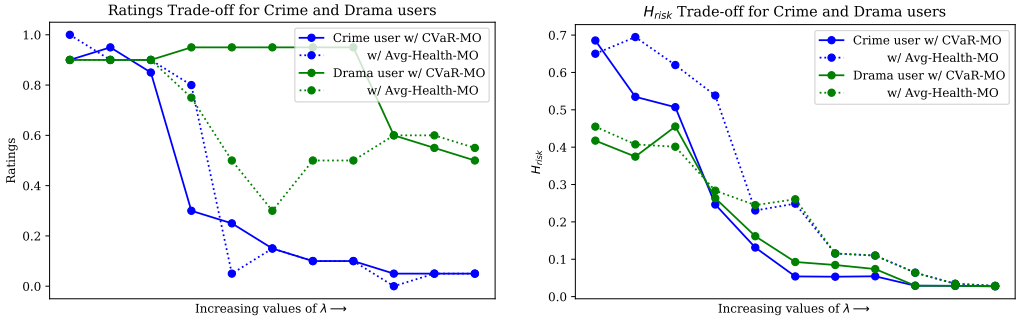

Fig. 6. Comparing the Ratings and Health risk trade-offs (respectively) for Crime and Drama genre users.

7 CONCLUSION AND FUTURE WORK

In this work, 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. We propose a Movielens based simulation environment to study this tradeoff and demonstrate that a purely rating optimizing agent could lead to unhealthy outcomes for 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. While our simulation setup is quite simplistic for this initial exploration, in future work, we expect to expand it's complexity, studying these phenomena on larger corpora and more complex user models.

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] 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 & Data Mining*. 2212–2220.

[3] 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 & development in information retrieval*. 405–414.

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

[5] 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.

[6] 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.

[7] 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 âĂŹ19)*. Association for Computing Machinery, New York, NY, USA, 456âĂŞ464. https://doi.org/10.1145/3289600.3290999

[8] 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

[9] 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

[10] 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.

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

[12] 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).

[13] 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.

[14] 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

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

[16] 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.

[17] 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.

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

[19] 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.

[20] 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.

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

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

[23] 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.

[24] 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).

[25] 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).

[26] 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.

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

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

[29] 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).

[30] 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.

[31] 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.

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

[33] 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.

[34] 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]

[35] 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.

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

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

[38] 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.

[39] 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.

[40] 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

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

[42] 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.

[43] 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.

[44] 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.

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

[46] Sai Wu, Weichao Ren, Chengchao Yu, Gang Chen, Dongxiang Zhang, and Jingbo Zhu. 2016. Personal recommendation using deep recurrent neural networks in NetEase. *2016 IEEE 32nd International Conference on Data Engineering (ICDE)* (2016), 1218–1229.

[47] 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).

[48] 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.