Towards Content Provider Aware Recommender Systems

A Simulation Study on the Interplay between User and Provider Utilities

Ruohan Zhan*, Konstantina Christakopoulou†, Ya Le†, Jayden Ooi†, Martin Mladenov†, Alex Beutel†, Craig Boutilier†, Ed H. Chi†, Minmin Chen†

rhzhan@stanford.edu; {konchris, elainele, jayden, mmladenov, alexbeutel, cboutilier, edchi, minminc}@google.com

*Stanford University, Stanford, CA

†Google, Mountain View, CA

ABSTRACT

Most existing recommender systems focus primarily on matching users (content consumers) to content which maximizes user satisfaction on the platform. It is increasingly obvious, however, that content providers have a critical influence on user satisfaction through content creation, largely determining the content pool available for recommendation. A natural question thus arises: can we design recommenders taking into account the long-term utility of both users and content providers? By doing so, we hope to sustain more content providers and a more diverse content pool for long-term user satisfaction. Understanding the full impact of recommendations on both user and content provider groups is challenging. This paper aims to serve as a research investigation of one approach toward building a content provider aware recommender, and evaluating its impact in a simulated setup.

To characterize the user-recommender-provider interdependence, we complement user modeling by formalizing provider dynamics as well. The resulting joint dynamical system gives rise to a weakly-coupled partially observable Markov decision process driven by recommender actions and user feedback to providers. We then build a REINFORCE recommender agent, coined EcoAgent, to optimize a joint objective of user utility and the counterfactual utility lift of the content provider associated with the recommended content, which we show to be equivalent to maximizing overall user utility and the utilities of all content providers on the platform under some mild assumptions. To evaluate our approach, we introduce a simulation environment capturing the key interactions among users, providers, and the recommender. We offer a number of simulated experiments that shed light on both the benefits *and the limitations* of our approach. These results help understand *how* and *when* a content provider aware recommender agent is of benefit in building multi-stakeholder recommender systems.

1 INTRODUCTION

Recommender systems have been playing an increasingly important role for online services by matching users with content for personalization and experience enhancement. Most recommender systems exclusively focus on serving content that maximizes user (consumer) satisfaction. Aligning a recommender system's objective with user utility is most natural given that users are first-hand consumers of recommendation services. However, another key group of players in any platform are the *content providers*, who are also largely influenced by the recommendation decisions. In a purely user-centric design, the exposure given to different content providers can be vastly disproportionate, with a small subpopulation receiving the majority of attention [22]. When this happens, less established content providers can find it difficult to break the popularity barrier and may eventually leave the platform due to lack of exposure [23]. This is commonly referred to in marketplaces as the 'superstar economy' [27], which in the long run can hurt the overall user utility by limiting the diversity of the pool of recommendable content.

As a result, there has been growing research interest in moving from a user-centric recommendation paradigm toward a multi-stakeholder setup [1, 10], under which the viability of content providers on the platform, as a function of their respective satisfaction, is also considered. Such content provider awareness can be achieved by maximizing user satisfaction with the constraint of keeping existing content providers on the platform [23], or by trading off content relevance with content provider fairness [22]. However, most existing work treats content providers as static entities. Here, we consider the dynamics of content providers in response to recommendation, and propose a reinforcement learning (RL)-based recommendation approach to optimize the combination of both user and provider satisfaction.

Optimizing a recommendation policy for content provider satisfaction is significantly more challenging than for user satisfaction for two main reasons: (i) lack of a clear definition of *provider utility*; and (ii) scalability. As opposed to more well-studied marketplaces, such as advertising platforms where suppliers clearly define their bidding policies, thus leaving signals indicating their utility [4], content provider utility on recommendation platforms has received little attention. In this paper, we consider one such definition: we assume that provider utilities can be manifested through their actions e.g., via their content uploading frequencies, their decision

Table 1: Related Work Comparison

	User Modeling	Content Provider Modeling	Counterfactual Reasoning	Joint User & Provider Optimization
Advertising Auctions [4, 12, 35]	No	Yes	Yes	No
Provider Fairness in RecSys [6, 8, 22, 23, 29]	Yes	No	Yes (in [22])	No
EcoAgent (This paper)	Yes	Yes	Yes	Yes

to stay/leave the platform, or more generally how they respond to recommender actions and user feedback.[1] This stands in contrast to existing work in fairness of recommender systems (e.g. [29] and references therein), where the direct influence of platforms/users on provider utility is not considered. Furthermore, in contrast to traditional two-sided marketplaces, such as [4], where advertiser bidding policies are more or less static, content providers in recommender systems tend to be more adaptive to the environment. For example, based on the popularity of their existing content, providers may adapt their next content focus. To capture such considerations, we model the provider dynamics as the counterpart of user dynamics in Markov decision processes and RL.

Second, while a single recommendation to a user only affects that specific user's utility, it can impact *all relevant content provider* utilities. For instance, the providers who are recommended may become more satisfied with the platform, while those who are not can have their satisfaction decreased, especially if their content has not been shown to users for a while. A recommendation objective that considers the utility of a single user and *all* content providers may be intractable on platforms with large numbers of providers. To address this, we propose an approximation under which the all-content-providers objective can be relaxed to just the recommended content provider's *utility lift*, defined as the difference in utility between the provider being recommended versus not.

We offer the following contributions:

- **RL for Content Provider Aware Recommendation**: We formulate recommendation as an RL problem for maximizing the joint objective of user and content provider utilities (Section 3). Building on the success of REINFORCE for user utility maximization [11], we propose a REINFORCE approach, coined EcoAgent (Section 4), on top of a **neural sequential representation** of both user and content provider trajectories to maximize our joint objective.
- **Counterfactual Content Provider Utility Uplift**: To address the computational concern of the effect of one recommendation on multiple content providers, we design a novel scalable objective that combines target user utility and recommended content provider utility *uplift*; and demonstrate that it is equivalent to optimizing overall user and content provider utilities under a mild condition (Section 3.4).
- **Multi-Stakeholder Simulation Environment**: To evaluate our approach in a setting where content providers' utilities are affected by both the users' actions and the recommender actions, we build a Gym [9] environment (Section 5). This environment aims at capturing a simplified version

of the user *and content provider* dynamics, which is not supported in most existing simulation platforms.[2]
- **Study of User vs. Content Provider Trade-offs**: We provide a series of experiments, each with the goal of identifying the extent to which our content provider aware RL recommender can positively affect user satisfaction, and the overall multi-stakeholder health, as measured by the user vs. content provider utility Pareto curve, and the number of viable providers on the platform (Section 6).

2 RELATED WORK

The impact of policies on content providers has been actively studied in two-sided markets with consumers and providers. There are two important lines of research: one takes the game theory perspective and develops econometric models for marketplaces such as advertising auctions [4, 12, 35]; another considers the fairness with respect to content providers in recommender systems [6, 8, 22, 23, 29]. Table 1 summarizes the comparison between these works and ours.

These two lines of work often have different modeling focuses on users and content providers, while our work aims at directly modeling their interaction dynamics with the platform. In the advertising auction literature, advertisers are modeled strategically [4, 12, 35]. User (consumer) utility is implicitly incorporated in advertiser valuations. A similar gaming perspective is also introduced to platforms with strategic content providers [5, 13, 14, 26]. Conversely, literature focusing on fairness of recommender systems with respect to content providers often assumes fully observed or static provider states, based on which user utility and provider fairness are jointly optimized [8, 22, 29]. However, as intelligent agents, content providers may have latent preferences/states influenced by users and recommender. To capture this, in contrast to prior work, we use Recurrent Neural Networks (RNNs) to model both user and content provider state transitions, and learn their sophisticated utility functions, which also implicitly account for their policies. With user and content provider characterizations in hand, we then update the agent through RL to optimize overall utilities.

The idea of using RL while utilizing temporal information in recommender systems has received increasing attention. There is an emerging literature in neural recommender systems that applies RNNs to encode user historical events to predict future trajectories [7, 15, 37]. Chen et al. utilize such a sequential representation as the user state encoding, on top of which an RL agent using a REINFORCE [36] policy-gradient-based approach is learned. Furthermore, the idea of jointly optimizing different objectives during

---

[1]Importantly, our experiments highlight the importance of investigating different provider utility definitions under different recommendation setups.

[2]Code of environment and RL algorithms can be found at https://github.com/google-research/google-research/tree/master/recs_ecosystem_creator_rl.

agent learning can be categorized into a broader literature on Multi-Objective Optimization [2, 25, 34].

Another difference of our work with the existing ones is the formulation of agent objective. In advertising auction literature, the goal is to construct an econometric model for a given marketplace, where advertiser strategies are inferred from the equilibria of the proposed model [4, 12]. The platform generates rankings that trade off user satisfaction, platform revenue, and advertiser profits [4]. We note a connection between advertiser strategies and our objective. A risk-neutral advertiser decides her bid by considering the incremental cost in an alternative ranking position [4, 12], while each content provider is considered by our EcoAgent based on their incremental (uplifted) utility resulted from one recommendation.

On the other hand, literature in fairness of recommender systems across content providers usually optimizes user satisfaction (ranking quality) as well as fairness with respect to content providers. The notion of fairness can be measured in multiple ways, such as content provider-groupwise model performance [6], equity of content provider attention [8], content provider popularity distribution [22], content provider exposure [29], etc. As noted by Mehrotra et al., such two objectives can be optimized through a combined metric of both (e.g., [6]) or constrained optimization, such as maximizing fairness with having an acceptable ranking quality [8], optimizing user utility while being fair to providers [29], etc. However, as pointed out by Asudeh et al., metrics of fairness need careful design and are sometimes chosen subjectively. In contrast, our work directly maximizes overall content provider utilities and implicitly considers disparity in content provider exposure and associated fairness in recommendations.

One closely related work featuring long-term 'social welfare' optimization in recommender system is proposed by Mladenov et al., wherein the authors optimize user utility while restricting matched content providers to be viable in equilibrium [23]. Our work differs in a few major ways. First and foremost, we learn fully dynamic recommendation policies which adapt to user feedback and provider behavior (e.g. changing topic focus) in every time step in contrast to a precomputed query-specific policy. Moreover, due to the partially-observed nature of our setting, the EcoAgent recommender naturally factors in issues such as exploration and uncertainty representation.

Finally, we relate our work to literature applying *counterfactual reasoning* to algorithm design, where the effect of a treatment (action, policy) is quantified by the difference of potential outcomes between the treated (with the treatment) and control (without the treatment) [28]. Usually, only one outcome is observed, and we need to infer the counterfactual alternatives. For example, Jaques et al. define an influence reward to be the treatment effect of an agent's action on other agents, and use it to increase agent-level collaboration in Multi-Agent RL [17]. Mehrotra et al. apply it to music recommendation platforms, where they analyze the treatment effect of one artist new track release on popularities of herself and other artists [21]. Similarly, in this paper, the content provider utility uplift is the treatment effect of current recommendation on content provider utility.
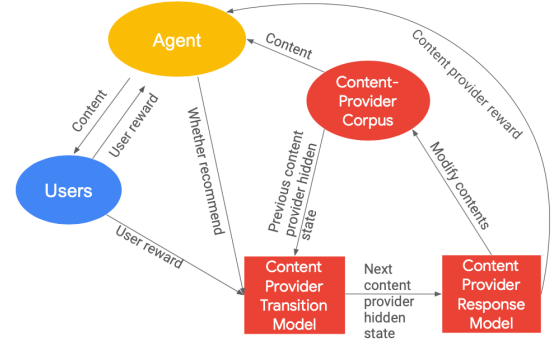

Figure 1: Interactions among Recommender Stakeholders

3 PROBLEM FORMULATION

We start by describing our problem setting, motivating the need for content provider aware RL-based recommenders and defining key terms to be used throughout.

3.1 RL for User Utility

A typical setup for long-term recommendation problems is as follows [11, 18, 30, 32]. A user issues a (possibly implicit) query to the recommendation system. The recommender agent responds with a slate of one or more items of content; the user interacts with the recommended content according to their (unobserved) preferences, emitting signals, such as likes, clicks, dwell times, indicating their engagement, satisfaction, etc. These proxy reward signals are then collected by the agent for future planning. We use *proxy reward* to refer to these immediate signals indicative of the user's affinity towards the recommended content, which can depend on topic preference alignment and content quality, among others.

We formulate the problem as a Markov decision process (MDP)[3] $(\mathcal{S}^u, \mathcal{A}, P^u, R^u, \rho_0^u, \gamma^u)$, where $\mathcal{S}^u$ is the state space encoding user topic preferences and context, $\mathcal{A}$ is the action space composed of the content available on the platform[4], $P^u : \mathcal{S}^u \times \mathcal{A} \to \Delta(\mathcal{S}^u)$ captures the state transition, R^u defines the user proxy reward, ρ_0^u is the initial state distribution, and γ^u the discount factor. A recommendation policy $\pi(\cdot|s)$ over the action space is learned to maximize the discounted cumulative reward across different user trajectories, which we refer to as the expected *user utility* under the policy π, that is

$$\max_{\pi} \quad \sum_{t=0}^{T} \mathbb{E}_{s_t^u \sim d_{\pi,t}^u(s_t^u), a_t \sim \pi(\cdot|s_t^u)} \left[Q_t^u(s_t^u, a_t) \right] \quad (1)$$

$$\text{where} \quad Q_t^u(s_t^u, a_t) = \sum_{t'=t}^{|\tau^u|} (\gamma^u)^{t'-t} r^u(s_{t'}^u, a_{t'}).$$

Here T is the maximum trajectory length, and τ^u is the user trajectory sampled according to the policy π under the user MDP. We use $d_{\pi,t}^u(\cdot)$ to denote the average user state visitation distribution

---

[3]Recommender systems are generally best formulated as *partially-observable MDPs* since user states are not observed. Here, for ease of exposition, we retain the MDP notation, assuming a learned state representation; but throughout (including experiments), we *infer* a user's state from their interaction history with the platform.

[4]W.l.o.g., we assume the agent recommends a single item to the user each time.

at time t [20], and $Q^u(s_t^u, a_t)$ is the user utility calculated from time step t according to the policy π. Note that the expectation takes into account different user trajectories induced by the learned policy.

3.2 Multi-Stakeholder Interactions

The above framing misses one group of key stakeholders: the content providers. To be able to optimize the overall utility of *all* stakeholders, it is necessary to extend the user-recommender interaction model to account for the impact of recommendations on providers' state and utility. Figure 1 gives an intuitive schema that captures stylized, yet real-world-inspired interactions among these stakeholders. As a simplification, we do not include environment externalities, e.g., exogenous discovery of provider.

Users ↔ Recommender Agent. The interactions between users and the recommendation agent are already captured in the setup of the previous section. One point to highlight is that not only does the agent adapt to user feedback, but users themselves are influenced by their interactions with the platform (e.g., a user's topic preferences might shift closer to or further from recommended content topics depending on how much they like the recommendations).

Content Providers ↔ Recommender Agent. Besides users, the agent also influences the content providers. This form of interaction has been largely ignored in the recommendation literature, but it is a key focus of this work. Specifically, depending on the exposure generated by the platform (i.e., how much the agent recommends a provider's content to users) and the user feedback (the users' *proxy reward* for such recommended content), content providers may decide to change the nature of their engagement with the platform. For example, content providers can decide to create more or less content, shift their topic focus, and or even leave the platform if their satisfaction falls below a certain point. This in turn influences the recommender agent, as it will directly affect the content corpus, from which recommendable candidate items are drawn.

3.3 RL for Content Provider Utility

Similar to a recommender agent that maximizes user utility as defined in Section 3.1, we can also define a MDP $(\mathcal{S}^c, \mathcal{A}, P^c, R^c, \rho_0^c, \gamma^c)$ for each content provider c. Here $\mathcal{S}^c$ is the content provider state space, $\mathcal{A}$ is again the action space composed of the content available on the platform, and $P^c : \mathcal{S}^c \times \mathcal{A} \rightarrow \Delta(\mathcal{S}^c)$ captures the content provider state transition, R^c defines the content provider proxy reward, ρ_0^c is the initial state distribution and γ^c the discount factor. We explain the content provider state, transition model and proxy reward in more detail.

Content Provider State s_t^c: The content provider state is influenced by the popularity of their current content, and captures the latent incentives for future content creation and preferences for content topics. In our design, we also force one component of the state to capture the content provider's current satisfaction with the platform, specifically the cumulative historical content provider proxy reward up to time t.

Content Provider State Transition $s_{t+1}^c \sim P^c(\cdot|s_t^c, a_t)$: Content provider states are changing over time based on feedback from the recommender and the users, i.e., how many recommendations the content provider received and how the users liked this content

provider's recommended content. For instance, if content provider's existing content is popular among users, the content provider's latent topic preference will shift towards existing content topics; otherwise they may change topics of future creation. The content provider satisfaction component of the state is incremented by the number of recommendations and summed user reward proxy signals acquired from the current time step.

Content Provider Reward Signals $r^c(s_t^c, a_t)$ **and Utility**: We use the content provider *proxy reward* r^c to capture content providers' incremental engagement/satisfaction with the platform at any time point. Such signals can manifest as viability (i.e., decide to stay/leave the platform), frequency of new content uploads, etc.[5] Conceptually, content provider reward captures how active the content provider is during a given time period. It also indicates how the content provider responds to *recommender and user feedback*. We argue that content providers of different groups will respond differently. Particularly, *less established content providers* who receive little attention or negative user feedback during certain time period are more likely to change their next content topics or become non-viable and eventually leave the platform, while the *established content providers* are more resilient to small changes in recommender and user feedback.

With the content provider MDP, we can learn a recommendation policy $\pi(\cdot|s)$ over the action space to maximize the discounted cumulative reward, which we refer to as the expected *content provider utility* under the policy π, that is

$$\max_{\pi} \quad \sum_{t=0}^{T} \mathbb{E}_{s_t^c \sim d_{\pi,t}^c(s_t^c), a_t \sim \pi(\cdot|s_t^c)} \left[Q_t^c(s_t^c, a_t) \right] \quad (2)$$

$$\text{where} \quad Q_t^c(s_t^c, a_t) = \sum_{t'=t}^{|\tau^c|} (\gamma^c)^{t'-t} r^c(s_{t'}^c, a_{t'}).$$

Here T is the maximum trajectory length, and τ^c is the creator trajectory sampled according to the policy π under the creator MDP. Similarly, we use $d_{\pi,t}^c(\cdot)$ to denote the average content provider state visitation distribution at time t [20], and $Q^c(s_t^c, a_t)$ is the content provider utility calculated from time step t according to policy π. Again the expectation takes into account the different content provider trajectories induced by the learned policy.

3.4 RL for Content Provider and User Utility

Having established that recommendations affect the future state and utility of both users *and content providers*, we formulate the recommendation problem as follows:

> *Decide which content (from which content provider) to recommend to a user so that a combined metric of both content provider and user utilities is maximized, given the current state of the user and the providers of candidate content recommendations.*

Putting everything together, we translate this setup into a user-and-content-provider MDP $(\mathcal{S}, \mathcal{A}, P, R, \rho_0, \gamma)$ where $\mathcal{S}$ is the concatenation of user state and states of all the content providers on the

---

[5]If we view content providers as rational decision-making agents, which could in turn be modeled as RL agents themselves, whether they will leave the platform, what type of content they create, etc., can be seen as content provider actions rather than reward proxy signals. Here though, we model this from the perspective of recommender as the RL agent finding the action so to maximize content provider–and user–utility.

platform, $\mathcal{A}$ is the content available on the platform, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ captures the state transition, R is the concatenation of proxy rewards of the user and all content providers, ρ_0 is the initial state distribution, and γ as concatenation of user and content provider discount factors. With this user-and-content-provider MDP, our goal is to learn a recommendation policy $\pi(\cdot|s)$ over the action space that maximizes two objectives: user utility and content provider utility. For simplicity, we adopt a scalarization approach, using a coefficient $\lambda \in [0, 1]$ to interpolate between the two objectives, which we refer to as the *content provider constant*; future work can look into other multi-objective optimization (MOO) approaches as well (e.g., [2, 25, 34]). When $\lambda = 0$, this is equivalent to RL for user utility (Section 3.1), whereas when $\lambda = 1$, this is a content provider-only optimizing policy (Section 3.3).[6]

$$\max_{\pi} \quad \sum_{t=0}^{T} \mathbb{E}_{\substack{s_t \sim d_{\pi, t}(s_t) \\ a_t \sim \pi(\cdot|s_t)}} \left[(1 - \lambda) Q_t^u(s_t^u, a_t) + \lambda \sum_{c \in C} Q_t^c(s_t^c, a_t) \right] \quad (3)$$

where T is the maximum trajectory length, C is the set of content providers on the platform[7], state $s_t = (s_t^u, s_t^{c_1}, \dots, s_t^{c_{|C|}})$ is the state concatenation of the user and all content providers, $d_{\pi, t}(s_t)$ is the average visitation distribution of s at time t [20], Q_t^u and Q_t^c are user and content provider utilities at time t defined in Eq. 1 and Eq. 2 respectively. Note that for each recommendation (user query), we consider the utility of the targeted user and the utilities of all relevant content providers in the action space—the content provider who is recommended increases her satisfaction, while others who are not have their satisfaction decreased.

Albeit complete, the objective as defined in eq. 3 is intractable on platforms with a large number of content providers. To make it tractable, we break down the utility of the content providers into the one whose content was recommended by the recommender and the others. That is,

$$\sum_{c \in C} Q_t^c(s_t^c, a_t) = Q_t^{c_{a_t}}(s_t^{c_{a_t}}, a_t) + \sum_{c' \in C \backslash c_{a_t}} Q_t^{c'}(s_t^{c'}, a_t) \quad (4)$$

Here we use c_{a_t} to denote the content provider associated with the chosen content a_t.

ASSUMPTION 1 (NO CONTENT PROVIDER EXTERNALITY). *A content provider's proxy reward and state transition of not being recommended do not depend on which provider is recommended instead.*

Intuitively, the above assumption describes that when a provider is not recommended, her reaction only depends on the fact that she was not exposed to users, but not on which other providers were recommended. We translate it to the following two equivalences: for any content provider $c' \in C \backslash c_{a_t}$ who is not chosen for current recommendation, we have

$$\text{proxy reward} \qquad r^c(s_t^{c'}, a_t) = r^c(s_t^{c'}, b_t^{c'})$$

$$\text{state transition} \qquad P^c(\cdot|s_t^{c'}, a_t) \stackrel{d}{=} P^c(\cdot|s_t^{c'}, b_t^{c'})$$

where $b_t^{c'}$ indicates any chosen content not associated with the content provider c' at time t. This further implies that, for any

content provider $c' \in C \backslash c^{a_t}$ who is not recommended at current time step, her utility later on does not depend on which other provider's content is recommended, i.e.,

$$Q_t^c(s_t^{c'}, a_t) = Q_t^c(s_t^{c'}, b_t^{c'}), \quad \forall c' \in C \backslash c^{a_t}. \quad (5)$$

Eq. 4 can then be simplified as the recommended content provider *utility uplift* plus a baseline as the sum of all content provider utilities when none of their content are recommended,

$$\sum_{c \in C} Q_t^c(s_t^c, a_t) \quad (6)$$

$$= \underbrace{Q_t^{c_{a_t}}(s_t^{c_{a_t}}, a_t) - Q_t^{c_{a_t}}(s_t^{c_{a_t}}, b_t^{c_{a_t}})}_{\text{utility uplift}} + \underbrace{\sum_{c' \in C} Q_t^{c'}(s_t^{c'}, b_t^{c'})}_{\text{baseline}}.$$

Note that the baseline has the merit of independence from the taken action and thus can be dropped when calculating the gradient of policy [36], which will be detailed in Section 4. As a result, the objective defined in eq. 3 can be simplified as optimizing user utility and the utility uplift of the recommended content provider,

$$\max_{\pi} \quad \sum_{t=0}^{T} \mathbb{E}_{s_t \sim d_{\pi, t}(s_t), a_t \sim \pi(\cdot|s_t)}[R_t] \quad (7)$$

$$\text{where} \quad R_t = (1 - \lambda) Q_t^u(s_t^u, a_t)$$
$$+ \lambda \left(Q_t^{c_{a_t}}(s_t^{c_{a_t}}, a_t) - Q_t^{c_{a_t}}(s_t^{c_{a_t}}, b_t^{c_{a_t}}) \right).$$

4 PROVIDER-AWARE REINFORCE AGENT

Now that we have formulated the multi-stakeholder recommender problem as an RL problem, we turn attention to learning the policy π that maximizes the objective Eq. 7. Different approaches [19, 24, 31, 33, 36] can be applied to solve this RL problem. Here, we adopt the policy-based REINFORCE [36] approach, inspired by its success in the classic user utility maximization recommender setting [11].

Let (parameterized) policy π_θ map states to actions. Using the log-trick, the policy parameters $\theta \in \mathbf{R}^d$ can be updated directly following the gradient,

$$\mathbb{E}_{s_t \sim d_{\pi, t}(s_t), a_t \sim \pi(\cdot|s_t)} [R_t \log \nabla_\theta \pi_\theta(a_t|s_t)]. \quad (8)$$

Eq. 8 gives an unbiased estimate of the policy gradient in online RL, where the gradient of the policy is computed on the trajectories collected by the same policy; this is our setup here.

In what follows, we outline how we parameterize the policy—particularly how to learn the user states (Section 4.1), the content provider states (Section 4.2), and the policy-specific parameters on top of these states to output probabilities over the action space (Section 4.3). Finally, we discuss in detail our reward, and specifically how we learn the content provider utility uplift part of the reward (Section 4.4), and close this section with an overview of how everything ties together to solve the multi-stakeholder recommendation RL problem (Section 4.5). We refer to our proposed learning approach as EcoAgent, illustrated in Figure 2.

4.1 User States via Utility Imputation

As described in Section 3.1, the user state $s^u \in \mathcal{S}^u$ encodes user preferences and context. To learn the user state, building on recent work showing the value of sequence learning [11], we use
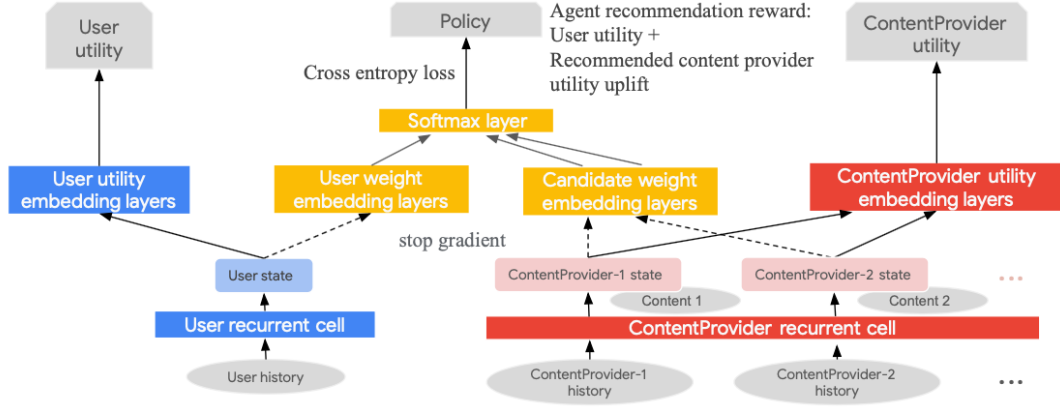
Figure 2: Illustration of EcoAgent structure. EcoAgent consists of three components: (i) a user RNN utility model that embeds user history into user hidden states and predicts user utility; (ii) a content provider RNN utility model that embeds content provider history into content provider hidden states and predicts content provider utility; (iii) an actor model that inputs user hidden state and candidates (content, content provider hidden state) to generate policy. Actor model is optimized using REINFORCE with recommendation reward being a linear combination of user utility and content provider utility uplift.

a recurrent neural network (RNN) to encode user interaction histories, and ensure the learned user state together with the action representation can predict observed user utility.

Let a user trajectory $\tau = \{(x_0^u, a_0, r_0^u), \cdots, (x_T^u, a_T, r_T^u)\}$ sampled according to the current policy π_θ, where x_t^u is user's context at time t. We can then further break τ into a set of history-action-return tuples of $\mathcal{D}^u = \{(H_t^u, a_t, Q_t^u)\}$ by aggregating the historical contexts and actions $H_t^u = \{(s_{t'}^u, a_{t'})\}_{t'=0}^t$ and accumulating the discounted future reward Q_t^u.

Given this collection $\mathcal{D}^u$ of historical contexts/actions and associated utilities, we can learn user utility model parameters ϕ so to minimize a loss function ℓ between the predicted and ground truth user utilities. This also offers the user hidden state s_t^u at time t:

$$\min_\phi \sum_{(H_t^u, a_t, Q_t^u) \in \mathcal{D}^u} \ell\left(\hat{Q}_t^u(s_t^u, a_t; \phi), Q_t^u\right) \quad (9)$$
$$\text{where } s_t^u = \text{RNN}(H_t^u).$$

In our experiments, we used Huber loss for ℓ.

4.2 Provider States via Utility Imputation

As described in Section 3.3, the content provider states encode content provider preferences and satisfaction. We learn the content provider states in similar fashion as the user states, "forcing" the state to be able to predict content provider utility.

Given a provider trajectory $\tau = \{(x_0^c, A_0, r_0^c), \cdots, (x_T^c, A_T, r_T^c)\}$, where x_t^c is content provider c's context at time t, we again break it into a set of history-action-return tuples of $\mathcal{D}^c = \{(H_t^c, A_t, Q_t^c)\}$ by aggregating historical contexts and actions, and accumulating the discounted future reward. In general, we expect content provider dynamics to evolve at a much slower rate than user dynamics; hence, the per-event time interval will typically be much different from that of the user trajectory. At each time step, the action $A_t = \{(v^{d_i}, r_i^u)\}_{i=1}^{m_t}$ records the provider's recommended item d_i and the user feedback r_i^u offered on this content, where m_t denotes

the number of recommendations made of this provider's contents, and v^{d_i} as an indicator vector of content d_i. Concretely, A_t is summarized by (i) the number of recommendations m_t this provider received at time t; (ii) the sum of user rewards $\sum_{i=1}^m r_i^u$; and (iii) a weighted bag-of-words representation of the recommended contents $\{d_i\}_{i=1}^{m_t}$, with weights reflecting the received user rewards: $\sum_{i=1}^{m_t} r_i v^{d_i} / \sum_{i=1}^{m_t} r_i$.

We employ another RNN to encode the historical actions into a content provider's state and learn the model parameters ψ so to minimize a loss function ℓ between the predicted and the ground truth content provider utilities,

$$\min_\psi \sum_{(H_t^c, A_t, Q_t^c) \in \mathcal{D}^c} \ell\left(\hat{Q}_t^c(s_t^c, A_t; \psi), Q_t^c\right) \quad (10)$$
$$\text{where } s_t^c = \text{RNN}(H_t^c).$$

4.3 Policy Parameterization

Conditioning on the user state s^u as learned by the User Utility model (Section 4.1) and each candidate content provider states s^c as learned by the content provider utility model (Section 4.2), the policy $\pi_\theta(a|s)$ mapping state to action is then learned with a simple softmax,

$$\pi_\theta(a|s) = \frac{\exp\langle\Phi(s^u), \Psi([a, s^{c_a}])\rangle/T)}{\sum_{a' \in \mathcal{A}} \exp(\langle\Phi(s^u), \Psi([a', s^{c_{a'}}])\rangle/T)}, \quad (11)$$

Note that this follows the paradigm of latent factor based recommendation where both users and candidate content items are mapped to the same low-dimensional embedding space, and the inner product $\langle,\rangle$ is used to capture the user-content proximity. Here, we use Φ to denote the projection of user state s^u to the embedding space. For the representation of the candidate content item, we concatenate the dense action/content embedding a with its associated content provider state s^{c_a}, and map them, via Ψ into the same low-dimensional space. T is a temperature term controlling the smoothness of the policy (set to 1 in our case).
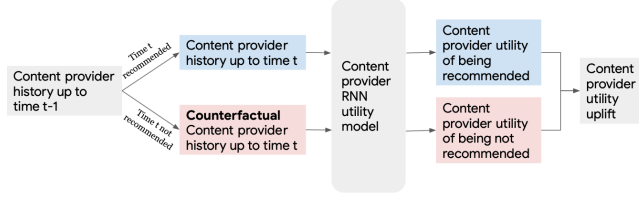
Figure 3: Illustration of Content Provider Utility Uplift.

4.4 Reward

Last, we need to specify our reward which will be guiding the learning of the policy. Recall that in eq. 7 we have defined our reward as the λ-weighted sum between the user utility and the content provider utility uplift.

User Utility Reward. Specifically, for user u and recommendation a_t at time t, we calculate user utility $Q_t^u(s_t^u, a_t)$ using the accumulated γ^u-discounted future reward of this user:

$$Q_t^u(s_t^u, a_t) = \sum_{t'=t}^{\infty} \left(\gamma^u\right)^{t'-t} r^u(s_{t'}^u, a_{t'}). \tag{12}$$

Content Provider Utility Uplift Reward. For the recommended content provider c^{a_t}, recall that this is defined as the difference of content provider factual utility when recommended versus the counterfactual utility when not recommended, as derived in eq. 6 and shown below,

$$Q_t^c(s_t^{c_{a_t}}, a_t) - Q_t^c(s_t^{c_{a_t}}, b_t^{c_{a_t}}) \tag{13}$$

However, for provider c_{a_t}, we do not have the data of her counterfactual utility supposing that she was not recommended at time t. We thus resort to content provider RNN utility model $\hat{Q}^c(s_t^c, A_t'; \psi)$ in Eq. 10 to predict her counterfactual utility value. Here A_t' stands for the counterfactual history where we remove the chosen action a_t from this content provider. To be robust to systematic prediction bias, we also use the utility model to estimate her factual utility for consistency. We illustrate this calculation in Figure 3.

4.5 Putting Everything Together

We adopt online learning for training EcoAgent and leave the discussion on offline learning to Section 7. In online RL, agent interacts with the environment to collect new data, based on which the agent is updated. We thus use the following optimization procedure:

(i) (EcoAgent in Action) Use current parametrized policy π_θ to interact with the environment and collect new data;
(ii) (EcoAgent in Optimization) Use newly-collected data to optimize the policy π_θ, and particularly the parameters of the user utility RNN model ϕ, the parameters of the content provider utility RNN model ψ, and the policy softmax parameters;
(iii) Repeat (i) and (ii) until converged.

5 SIMULATED ENVIRONMENT

We opt for a simulation environment to study the effectiveness of our approach for two reasons. First, publicly available recommender datasets do not consider the content provider aspect of the recommendation problem; they therefore cannot serve as a

basis for evaluating RL for multi-stakeholder recommendation. Second, simulation allows us to control the underlying environmental assumptions and study their effects on the performance of our proposed approach.

Although there has been prior work on simulation recommendation environments [23], it does not capture the full spectrum of users-recommender-content providers interactions described in Section 3. We thus implemented a new Gym [9] environment on top of RecSim which is a simulation platform with user dynamics [16]. This environment is designed to capture the content provider dynamics as illustrated in Figure 1. We now offer more detail on the underlying dynamics used in our experiments.

5.1 Content Description

We assume that there are K content *topics*. Each content item d has an observable one-hot vector $v^d \in \{0, 1\}^K \sim \text{Categorical}(p)$ representing its topic and a scalar *quality* $q^d \sim \text{Truncated normal}(\mu, \sigma, -1, 1)$ that is perceived by the user but is hidden to the agent. Both p and (μ, σ) are content provider-dependent, with p reflecting a provider's topic preference of future creation, which can shift during the provider's interaction with the platform.

5.2 User Updates

Each user u is described by a unit vector $v_t^u \in \mathbb{R}^K$ that represents her topic preference at time t. The initial state v_0^u is sampled from a uniform distribution over the unit ball. After being recommended some item, the user immediate reward r_t^u is a linear combination of the content's relevance (defined as the inner product of user topic preference v_t^u and content topic v^d) and content quality q^d, with η^u denoting the user's sensitivity to content quality: $r_t^u = (1 - \eta^u)\langle v^d, v_t^u \rangle + \eta^u q^d$. User topic preference then shifts toward the topic of the recommended content, weighted by how much the user liked it, r^u; thus $v_{t+1}^u \leftarrow v_t^u + \delta^u r^u v^d$, with δ^u denoting the degree to which the user's topic preferences is influenced by the recommended content.

5.3 Content Provider Updates

Each content provider c is described by a vector $v_t^c \in \mathbb{R}^K$ that represents her topic preference for future creation at time t. The initial state v_0^c is sampled from a uniform distribution over the unit ball. Each provider starts with a fixed number of content items, but the amount of content can increase over time if she decides to create more.

Suppose that at time t, provider c receives m recommendations of her content $(d_1, \ldots, d_m)$ and user rewards $(r^{u_1}, \ldots, r^{u_m})$. We aggregate the recommender-induced exposure and user feedback as

$$p_t^c = \mu^c + \eta_1^c m + \eta_2^c \sum_{i=1}^{m} r^{u_m}, \tag{14}$$

where η_1^c, η_2^c represent content provider sensitivity to content exposure and user feedback respectively, and $\mu^c < 0$ accounts for the *negative impact of no recommendation*. Provider satisfaction at time t is then defined as

$$S_t^c = f\left(\sum_{s=0}^{t} p_s^c\right). \tag{15}$$

Here f transforms the raw feedback into content provider satisfaction. Different choices of f can represent different content provider characteristics, which we detail below.

Content provider reward is then defined to be incremental provider satisfaction:

$$r_t^c = S_t^c - S_{t-1}^c. \tag{16}$$

Specifically, when f is the identity, the provider reward is simply the raw feedback p_t^c. If her current reward r_t^c is positive, the provider will create more content, where the number of new items is proportional to r_t^c.

To model content providers adapting their topic preferences based on user feedback, a content provider updates her topic preference using the sum of her recommended content topics, weighted by the corresponding user feedback:

$$v_{t+1}^c \leftarrow v_t^c + \delta^c \sum_{i=1}^{m} r^{u_i} v^{d_i}, \tag{17}$$

where δ^c represents how sensitive the content provider's topic preferences are to the user feedback.

Each content provider also has a viability threshold; if their satisfaction S_t^c is below the threshold, they leave the platform.

Content Provider Satisfaction Design. As mentioned, content provider dynamics are characterized by the function f, transforming the raw recommender and user feedback to content provider satisfaction. While a linear transformation is the most intuitive, concave functions such as logarithm (see Figure 4a), reflect the law of diminishing returns often seen in real life. For example, as discussed in Section 3.3, one recommendation may not influence established content providers much given their existing popularity base. In contrast, for less established providers who are considering leaving the platform due to insufficient prior exposure, a new recommendation may change their mind and thus help the platform retain them. With a concave satisfaction function, less established content providers will derive more utility than more established providers given the same amount of raw recommender and user feedback p_t^c. We often adopt this general assumption:

ASSUMPTION 2 (SATURATED CONTENT PROVIDER SATISFACTION). *Content provider satisfaction gets saturated with accumulated recommender and user feedback.*

6 EXPERIMENTS

After defining our simulation environment, we now turn to our experiment results.

6.1 Experiment Setup

During online learning, EcoAgent collects data by interacting with a new environment with prespecified hyperparameters characterizing user and content provider dynamics described in Section 5. Each new environment samples initial states of 50 users and 10 content providers uniformly from the state spaces. Each content provider is initialized with 20 content items which are sampled from 10 topics based on provider's topic preference[8], and they have the choice

---

[8]We design our environment to have more users than content providers, and the number of items is larger than the number of users, following the statistics on current content sharing platforms.
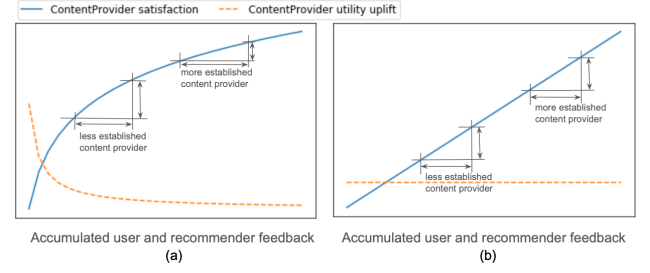
Figure 4: Two configurations for content provider satisfaction function f over accumulated recommender and user feedback: (a) log function, saturating as recommender and user feedback gets increased, (b) linear function, where there is no difference of recommendation effect on content provider utilities between a less established content provider versus an established content provider.

of creating more content or leaving the platform as described in Section 5. New comers are not considered.

For each training epoch, EcoAgent interacts with 10 new environments as set up above. The environment will be rolled out for 20 steps. At each time step of one rollout, all users receive recommendations simultaneously, and the environment updates all users' and content providers' states. We then use collected data to update EcoAgent with Adagrad optimizer. Throughout the experiments, we consider long-term effects on users and content providers by setting discount factors: $\gamma^u = \gamma^c = 0.99$.

6.1.1 Baselines. For each environment setup (user and content provider dynamics), we consider EcoAgents with different λ's (content provider constant) varying from 0 to 1. We highlight three baselines:

- **user-only EcoAgent**, where $\lambda = 0$. EcoAgent only optimizes user utility;
- **content provider-only EcoAgent**, where $\lambda = 1$. EcoAgent only optimizes content provider utility;
- **random agent**, where the agent recommends content randomly from the candidate set of content items.

Conceptually, user-only EcoAgent should have the largest user satisfaction, while content provider-only EcoAgent has the largest content provider satisfaction. The random agent sets a baseline on how EcoAgent learns user preference as well as on how *content provider aware* ($\lambda > 0$) EcoAgents help content providers.

For every agent considered, we use LSTM cells in utility models with $(32, 32, 16)$ hidden layers between LSTM output and utility prediction, which are chosen by preliminary experiments to minimize the loss of utility prediction. For actor model, we first embed user states and candidate (content provider state, content) tuples into weights using $(32, 32, 32)$ fully-connected layers (again tuned by preliminary experiments), then generate policy based on the softmax of the dot product between user weight and candidate weights. We tune EcoAgents with various learning rates and select the model that maximizes the objective function eq. 7 with corresponding content provider constant λ.
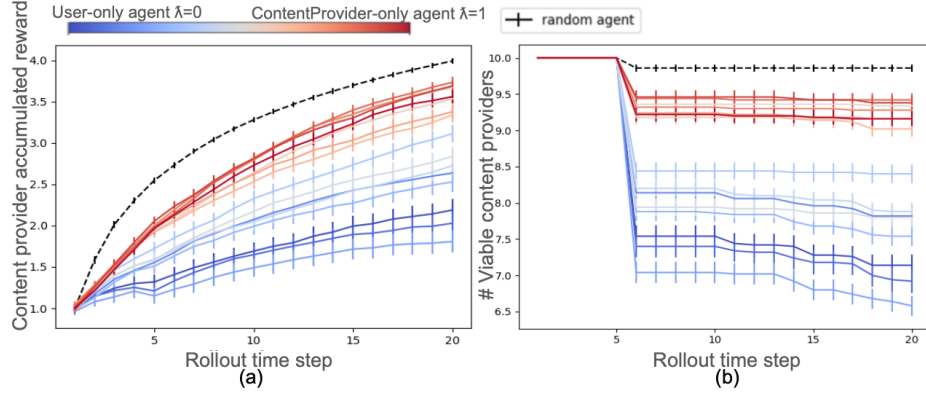
Figure 5: A content provider oriented EcoAgent (λ close to 1) helps content providers by improving content provider accumulated reward and number of viable content providers as compared to a user-oriented EcoAgent (λ close to 0).

6.1.2 Metrics. To compare different agent models, we test them in 50 rollouts of new environments for 20 steps (with prespecified user and content provider dynamics but different initial states). We calculate the following statistics of each rollout, which summarize how the agent influences users and content providers respectively:

- **user accumulated reward**: $\sum_{t=1}^{20} r_t^u$;
- **content provider accumulated reward**: $\sum_{t=1}^{20} r_t^c$;
- **# viable content providers**: number of providers in the environment at the current time step.

Both accumulated rewards characterize how satisfied users and content providers are on the platform, and number of viable providers reflects how agents help less established providers. In all the plots below, we show the average and the standard error (error bar) of these statistics across test rollouts.

6.2 When, and Why, EcoAgent helps both Provider and User Utilities?

We start with presenting our results under the main environment setup – *saturated content provider satisfaction*, which defines content provider satisfaction following Assumption 2.

Does EcoAgent increase content provider satisfaction? The first question is if the consideration of content provider utility uplift can indeed lead to happier content providers. We vary the content provider constant λ from 0 (user-only EcoAgent) to 1 (content provider-only EcoAgent) as shown in Figure 5. It can be seen that content provider accumulated reward (Figure 5a) and number of viable content providers (Figure 5b) are increased with content provider oriented EcoAgents (λ close to 1) as compared to user-oriented EcoAgent (λ close to 0). Particularly, content provider-only EcoAgent ($\lambda = 1$) generates recommendations only to maximize content provider utility and thus has largest content provider satisfaction as well as most viable content providers.

Recall that content provider reward is decided by number of recommendations and user feedback on the recommended content. Figure 6a shows the percentage of being recommended contributing to provider reward, and Figure 6b shows the total user feedback
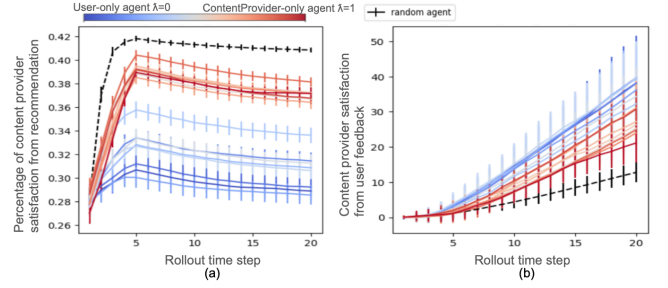
Figure 6: Decomposition of content provider reward with respect to being recommended and user feedback. (a) With a content provider aware EcoAgent, being recommended contributes more to provider utilities as compared to that with a user-only agent. (b) The user-feedback part of the reward is higher for EcoAgents compared to a random agent, as the latter ignores user preference.
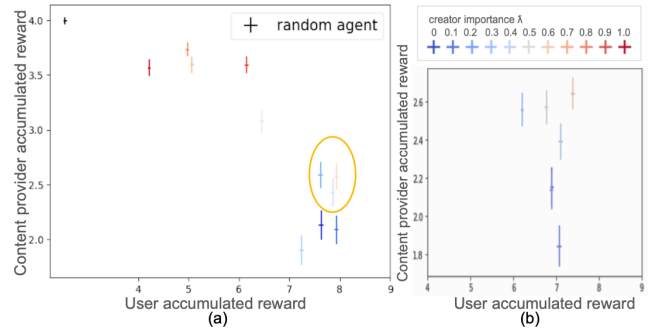
Figure 7: Tradeoff between users and content providers accumulated reward. (a) A properly tuned content provider aware EcoAgent can have a good balance between user utility and content provider utility. (b) In a setup where there exist two identical groups of providers, differing only in initial satisfaction, as expected, there is no tradeoff—we can increase provider utility without sacrificing user utility.
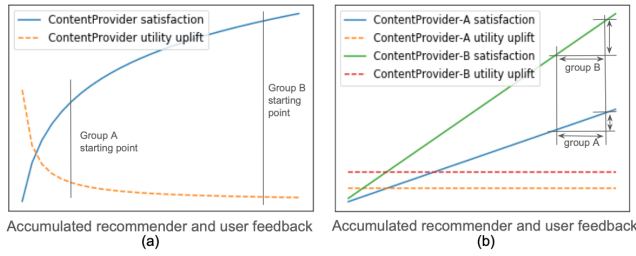
Figure 8: Illustration of environment setups with subgroup content providers. Figure (a) shows an environment of saturated content provider satisfaction, which is initialized with two identical content provider subgroups except that they have different starting points for satisfaction. Figure (b) shows an environment of linear content provider satisfaction, initialized with two identical content provider subgroups except that they have different rate in increasing satisfaction with incremental recommender and user feedback.

contributing to provider reward. With a random agent, recommendations are more evenly distributed among content providers, which results in a noticeable portion of provider reward coming from being recommended (Figure 6a); larger as compared to EcoAgents. However, a random agent does not consider user preference, and thus the user-feedback portion of provider reward is significantly lower than EcoAgents as demonstrated in Figure 6b.

Will user satisfaction be sacrificed? The second question comes naturally given that EcoAgent improves content provider satisfaction—then how about users? As the Pareto curve in Figure 7a shows, a heavily content provider aware EcoAgent (λ close to 1), though improving much on content provider accumulated reward, has noticeably smaller user accumulated reward as compared to a user-only EcoAgent, which is yet still better than a random agent. However, EcoAgents in the yellow circle of Figure 7a suggests that, for a properly content provider aware EcoAgent ($0 < \lambda \leq 0.6$), we can improve content provider satisfaction without sacrificing user satisfaction much. What is remarkable is that, with certain λ parameter choices, not only content provider reward is increased, but also user reward is higher compared to a user-only agent ($\lambda = 0$).

To gain further understanding on the trade-off between user and provider utilities, we consider an additional environment setup where there are two provider groups, who have identical features except for the initial content provider satisfaction—Group A has a smaller starting point, while group B has a larger starting point, as shown in Figure 8a. Conceptually, group A should be recommended more by EcoAgent, since doing so brings more content provider utility uplift; but this should not impact user satisfaction much, since these two groups are identical except the initial content provider satisfaction. Figure 7b verifies our conjecture: these EcoAgents with $\lambda \leq 0.6$ increase provider accumulated reward without sacrificing user accumulated reward, that is, a content provider aware EcoAgent with an appropriate provider constant can achieve a good balance between user and content provider satisfactions, verifying the promise of the approach in this setting.
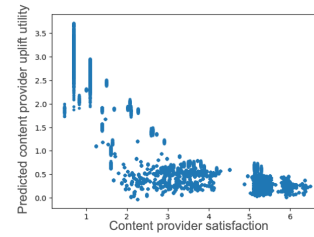
Figure 9: Content provider utility uplift versus content provider satisfaction. In an environment with log (and thus saturated) content provider satisfaction shown in Figure 4a, predicted content provider utility uplift by EcoAgent is inversely proportionally to content provider satisfaction. EcoAgent promotes less established content providers more to gain more content provider utility uplift, which thus increases overall content provider utilities.

Why does content provider utility uplift help? We next examine the rationale behind content provider utility uplift. In an environment with saturated content provider satisfaction, the predicted content provider utility uplift by EcoAgent is inversely proportional to content provider satisfaction as shown in Figure 9. In such cases, less established content providers have larger ground truth content provider utility uplift as compared to established content providers (Figure 4a). By predicting content provider utility uplift, EcoAgent can identify and promote those less established content providers who need recommendation most, which increases overall content provider utilities and also helps platform have more content providers viable, as we already discussed in Figure 5.

6.3 When, and why, EcoAgent does *not* lead to a healthier system?

Until this point, one can clearly see that EcoAgent helps content providers, as long as content provider satisfaction gets saturated. It raises a natural question on how sensitive EcoAgent is to environmental setup and what happens in the opposite scenarios.

What if content provider satisfaction does not get saturated? We consider an environment with *linear content provider satisfaction*. As opposed to saturated content provider satisfaction function f such as logarithm, here we consider f to be linear, where content provider satisfaction S_t^c and accumulated recommender and user feedback $\sum_{s=1}^{t} p_t^c$ are exactly in proportion: $S_t^c = f(\sum_{s=1}^{t} p_s^c) = \eta \sum_{s=1}^{t} p_s^c$, as demonstrated in Figure 4b. Content provider utility uplift thus remains the same across different content providers. As expected, EcoAgent would not help here as shown in Figure 10, since there is no guidance for EcoAgent to distinguish less established content providers from established content providers.

Is there any setup that content provider aware EcoAgent performs differently to a user-only EcoAgent, under the linear content provider satisfaction setting? The answer is positive as long as content providers with different satisfaction have different increasing rates with respect to accumulated recommender and user feedback. This allows EcoAgents to identify and recommend content providers with larger satisfaction increasing rate, which
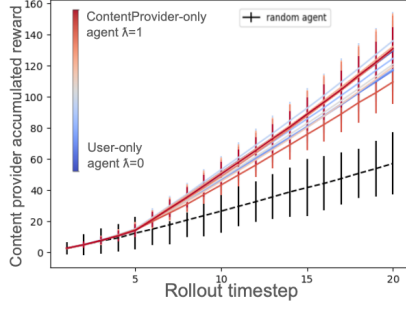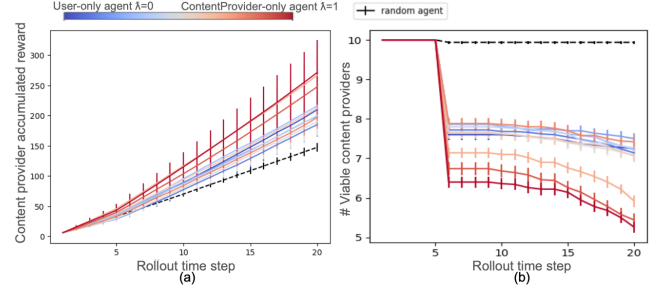
Figure 10: Performance of EcoAgents when content provider satisfaction does not get saturated. In an environment of linear provider-satisfaction shown in Figure 4b, EcoAgents with varying λ (content provider constant) achieve similar results on provider accumulated reward. This is because there is no difference on provider uplift utilities among different providers, so that EcoAgents cannot identify less established providers versus established providers.

brings more provider utility uplift. To achieve this, we again consider two identical provider groups with linear provider satisfaction, expect that the slope of each provider satisfaction function differs:

- group A content providers: $S_t^{c_A} = \eta_A \sum_{s=1}^{t} p_t^{c_A}$;
- group B content providers: $S_t^{c_B} = \eta_B \sum_{s=1}^{t} p_t^{c_B}$,

where $\eta_B > \eta_A$ such that group B content providers increase their utilities faster than group A content providers, as shown in Figure 8b. By design, EcoAgent can identify these two groups based on predicted content provider utility uplift—group B should have larger uplift. Figure 11a shows that content provider aware EcoAgents increase content provider satisfaction as compared to a user-only EcoAgent. This is because EcoAgents can gain more content provider utility uplift by promoting group B more (shown in Figure 12), thus improving overall content provider satisfaction.

However, does EcoAgent really help keep many viable content providers this time? Surprisingly, Figure 11b shows that the number of viable content providers with content provider aware EcoAgents instead decreases, even though these EcoAgents have larger content provider accumulated reward as compared to a user-only EcoAgent (Figure 11a). The problem originates from the incapability of utility uplift to capture the difference between less established content providers and established content providers, when the saturated satisfaction is violated. This leads to our reflection on EcoAgent objective, which uses content provider utility uplift to optimize overall content provider utilities. Such objective is rational as long as content provider utility can fully characterize the happiness of each content provider individual, or the platform fairness with respect to content providers. However, *if one's ultimate goal is to maintain a healthy multi-stakeholder platform of more content providers and users, only resorting to maximizing content provider utility can be biased*. In such cases, EcoAgent objective needs to be adjusted correspondingly, such as including *content provider viability uplift*—the difference of content provider viability probabilities between being recommended versus not.

Figure 11: Performance of EcoAgents for subgroups of content providers with different linear satisfaction functions. Consider an environment with subgroups of content providers who have different rates in increasing satisfaction with respect to accumulated recommender and user feedback, as shown in Figure 8(b). (a) EcoAgent improves content provider accumulated reward as compared to a user-only agent. (b) However, this reduces number of viable content providers on the platform, which results from the EcoAgent's unfair favor of those content providers who are more sensitive to environment but otherwise have no difference from the others.
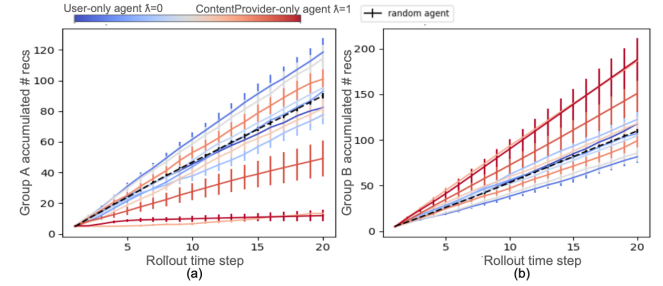
Figure 12: Accumulated recommendations of content provider subgroups A and B of Figure 8(b): (a) for group A with the slower satisfaction increasing rate, (b) for group B, with the faster satisfaction increasing rate. EcoAgents with higher λ (content provider constant) learn to recommend group B more, since doing so results in larger provider utility uplift; this in turn leads to more group B viable providers, but fewer group A viable providers—with a decreased number of total (group A+B) viable providers.

7 DISCUSSION

In this paper we study a content provider aware recommender agent that aims at maximizing the combined user and content provider utilities, which we refer to as EcoAgent. We show that maximizing the utility uplift of the provider associated with the chosen content is equivalent to maximizing utilities of all relevant content providers under a mild condition, which merits the scalability of our algorithm. To evaluate EcoAgent, we develop a Gym environment that allows for complex interactions among users, content providers, and agent. We conducted a series of experiments

to identify scenarios under which a content provider aware RL recommender can lead to longer term user satisfaction and healthier multi-stakeholder recommender systems.

A number of important research directions remain open. For example, how to adapt EcoAgent to optimize different metrics, such as number of viable providers or the number of content items. One can modify the lift measurement of utility to alternatives such as modeling the lift of viability probability or content uploading frequencies. Another interesting direction is to consider interference among providers, which violates the no-content-provider-externality Assumption 1.

Together, we hope that this study can motivate future research in understanding the impact of recommendation on different entities on the recommendation platforms and building healthy multi-stakeholder recommender systems.

REFERENCES

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Recommender systems as multistakeholder environments. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. 347–348.

[2] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2011. Click shaping to optimize multiple objectives. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 132–140.

[3] Abolfazl Asudeh, HV Jagadish, Julia Stoyanovich, and Gautam Das. 2019. Designing fair ranking schemes. In *Proceedings of the 2019 International Conference on Management of Data*. 1259–1276.

[4] Susan Athey and Denis Nekipelov. 2010. A structural model of sponsored search advertising auctions. In *Sixth ad auctions workshop*, Vol. 15.

[5] Omer Ben-Porat and Moshe Tennenholtz. 2018. A game-theoretic approach to recommendation systems with strategic content providers. In *Advances in Neural Information Processing Systems*. 1110–1120.

[6] 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.

[7] Alex Beutel, Paul Covington, Sagar Jain, Can Xu, Jia Li, Vince Gatto, and Ed H Chi. 2018. Latent cross: Making use of context in recurrent recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 46–54.

[8] 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.

[9] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. arXiv:arXiv:1606.01540

[10] Robin D Burke, Himan Abdollahpouri, Bamshad Mobasher, and Trinadh Gupta. 2016. Towards Multi-Stakeholder Utility Evaluation of Recommender Systems.. In *UMAP (Extended Proceedings)*.

[11] 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*. 456–464.

[12] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. 2007. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American economic review* 97, 1 (2007), 242–259.

[13] Arpita Ghosh and Patrick Hummel. 2014. A game-theoretic analysis of rank-order mechanisms for user-generated content. *Journal of Economic Theory* 154 (2014), 349–374.

[14] Ashish Goel and Farnaz Ronaghi. 2012. A game-theoretic model of attention in social networks. In *International Workshop on Algorithms and Models for the Web-Graph*. Springer, 78–92.

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

[16] 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).

[17] Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, DJ Strouse, Joel Z Leibo, and Nando De Freitas. 2019. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International Conference on Machine Learning*. PMLR, 3040–3049.

[18] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. 781–789.

[19] Sergey Levine and Vladlen Koltun. 2013. Guided policy search. In *International Conference on Machine Learning*. 1–9.

[20] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643* (2020).

[21] Rishabh Mehrotra, Prasanta Bhattacharya, and Mounia Lalmas. 2020. Inferring the Causal Impact of New Track Releases on Music Recommendation Platforms through Counterfactual Predictions. In *Fourteenth ACM Conference on Recommender Systems*. 687–691.

[22] Rishabh Mehrotra, James McInerney, Hugues Bouchard, Mounia Lalmas, and Fernando Diaz. 2018. Towards a fair marketplace: Counterfactual evaluation of the trade-off between relevance, fairness & satisfaction in recommendation systems. In *Proceedings of the 27th acm international conference on information and knowledge management*. 2243–2251.

[23] 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. *arXiv preprint arXiv:2008.00104* (2020).

[24] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[25] Hossam Mossalam, Yannis M Assael, Diederik M Roijers, and Shimon Whiteson. 2016. Multi-objective deep reinforcement learning. *arXiv preprint arXiv:1610.02707* (2016).

[26] Nimrod Raifer, Fiana Raiber, Moshe Tennenholtz, and Oren Kurland. 2017. Information retrieval meets game theory: The ranking competition between documents' authors. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 465–474.

[27] Sherwin Rosen. 1981. The economics of superstars. *The American economic review* 71, 5 (1981), 845–858.

[28] Donald B Rubin. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology* 66, 5 (1974), 688.

[29] 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.

[30] Alex Strehl, John Langford, Lihong Li, and Sham M Kakade. 2010. Learning from logged implicit exploration data. In *Advances in neural information processing systems*. 2217–2225.

[31] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*. 1057–1063.

[32] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miro Dudik, John Langford, Damien Jose, and Imed Zitouni. 2017. Off-policy evaluation for slate recommendation. In *Advances in Neural Information Processing Systems*. 3632–3642.

[33] Hado Van Hasselt, Arthur Guez, and David Silver. 2015. Deep reinforcement learning with double q-learning. *arXiv preprint arXiv:1509.06461* (2015).

[34] Kristof Van Moffaert, Madalina M Drugan, and Ann Nowé. 2013. Scalarized multi-objective reinforcement learning: Novel design techniques. In *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*. IEEE, 191–199.

[35] William Vickrey. 1961. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance* 16, 1 (1961), 8–37.

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

[37] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent recommender networks. In *Proceedings of the tenth ACM international conference on web search and data mining*. 495–503.