Sequential Bayesian Search

Zheng Wen zhengwen@stanford.edu
Electrical Engineering Department, Stanford University, Stanford, CA 94305, USA

Branislav Kveton Branislav.Kveton@technicolor.com
Technicolor Research Center, 735 Emerson St, Palo Alto, CA 94301, USA

Brian Eriksson Brian.Eriksson@technicolor.com
Technicolor Research Center, 735 Emerson St, Palo Alto, CA 94301, USA

Sandilya Bhamidipati Sandilya.Bhamidipati@technicolor.com
Technicolor Research Center, 735 Emerson St, Palo Alto, CA 94301, USA

Abstract

Millions of people search daily for movies, music, and books on the Internet. Unfortunately, non-personalized exploration of items can result in an infeasible number of costly interaction steps. We study the problem of efficient, repeated interactive search. In this problem, the user is navigated to the items of interest through a series of options and our objective is to learn a better search policy from past interactions with the user. We propose an efficient learning algorithm for solving the problem, sequential Bayesian search (SBS), and prove that it is Bayesian optimal. We also analyze the algorithm from the frequentist point of view and show that its regret is sublinear in the number of searches. Finally, we evaluate our method on a real-world movie discovery problem and show that it performs nearly optimally as the number of searches increases.

1. Introduction

Millions of people search daily for movies, books, and music on the Internet. Existing recommender systems, like Netflix and Pandora, typically allow their users to browse their collections of content in some predefined taxonomy, from general item categories, such as movie genres, to more specific ones. When the users look for items in obscure and niche categories, they are usually hard to find. This is unfortunate since such items are often of the highest value. The ability to find relevant content fast is important, and would ultimately lead to better recommendations, user experience, and content monetization.

In modern recommender systems, the number of item categories is often large, and they can be very general or specific. In the movie recommender system Netflix, the categories range from *Romance* to *Nature Family-Time TV*. In the music recommender system Pandora, the categories range from *Classical* to *East Coast Hip Hop*. To navigate through these categories, the user is first asked to choose a broad area, such as *Drama* or *Comedy*, and then chooses from more specific options, such as *Independent* or *Romantic Drama*. This policy for navigating the space of content can be represented by a tree, where the nodes and branches are the user's options and choices, respectively. We refer to this tree as a *taxonomic tree*.

In this work, we study how to learn better taxonomic trees over time based on the user's behavior. Suppose that the user enjoys *Cult Horror*. This movie genre is not widely popular. So in a taxonomic tree that is not personalized, the user would have to navigate through many options before getting to relevant movies. If the tree was personalized, *Cult Horror* could be one of the options at the root, and therefore the user would find relevant content faster.

Our learning problem is a repeating interactive game, which comprises of *episodes*. In each episode, we navigate the user to the item of interest by asking a series of *questions*. The questions depend on searched items in the past episodes and the answers to the questions in the current episode. We interact with the user until we can identify a single *target item*, the item of user's interest. Then we update our estimator of user pref-

Proceedings of the 30^{th} International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

erences $\mathbb{P}_t$ and the game proceeds to the next episode. Assuming that the user repeatedly searches for similar content, we expect to learn a better policy for content search over time.

We solve our problem by Bayesian learning and hence we refer to our solution as *Sequential Bayesian Search (SBS)*. Our solution has several nice properties. First, it is *computationally efficient*. In particular, we show that the posterior $\mathbb{P}_t$ over user preferences π^* can be updated by counting. Even more importantly, nearly optimal policies for finding items with respect to $\mathbb{P}_t$, a distribution over user preferences, can be derived as efficiently as with respect to any particular preference π. Second, we show that our solution is *Bayesian optimal*. Third, we analyze it from the frequentist point of view and prove that its *regret*, additional questions asked due to not knowing the true user preferences, is $O(\sqrt{t \ln(t)})$, *sublinear* in the number of episodes t. Finally, we evaluate our solution on a movie discovery problem and show how it improves with training episodes.

This paper is organized as follows. In Section 2, we introduce the notion of interactive search. In Section 3, we discuss related work. In Section 4, we propose sequential Bayesian search, discuss how to implement it, and show its Bayesian optimality. In Section 5, we analyze our approach from the frequentist point of view and bound its regret. In Section 6, we evaluate it on a real-world problem. Finally, we conclude in Section 7.

2. Problem Setup and Notation

In this section, we discuss how the problem of content discovery can be viewed as a game, where the content discovery system asks the user a sequence of questions to find the target item of interest. We assume that the user has a *target item* i^* in mind. This item belongs to a set of M *items* $\mathcal{I} = \{1, \cdots, M\}$. The *preferences* of the user for the items are modeled by a probability distribution π^* over $\mathcal{I}$ and the target item i^* is drawn i.i.d. from this distribution.

Our goal is to identify the item i^* by asking the user a sequence of yes-or-no questions q. The *questions* are modeled as functions $q : \mathcal{I} \to \{0, 1\}$ such that $q(i) = 1$ when the item i has the attribute q and $q(i) = 0$ when it does not. The set of all questions is referred to as the *question pool* $\mathcal{Q}$. We assume that all items can be uniquely identified by a subset of questions from $\mathcal{Q}$.

The search for the target item i^* proceeds as follows. First, we ask a question q_1 and the user replies $q_1(i^*)$. Second, we ask another question q_2 and get an answer $q_2(i^*)$. Note that the choice of the second question q_2 may depend on the answer $q_1(i^*)$ to the first question. In general, at time ℓ we choose a question q_ℓ based on all questions and answers up to time ℓ, $(q_j, q_j(i^*))_{j=1}^{\ell-1}$. Formally, a *policy* for choosing questions is a function:

$$T : 2^{\mathcal{Q}} \times \{0, 1\}^{|\mathcal{Q}|} \to \mathcal{Q}, \tag{1}$$

where $2^{\mathcal{Q}}$ is the index set of the question pool $\mathcal{Q}$ and $\{0, 1\}^{|\mathcal{Q}|}$ is a vector of $|Q|$ answers. The policy can be represented by a *decision tree* T, where the nodes and branches correspond to the asked questions and user's answers, respectively (Golovin & Krause, 2011).

The search continues until only one item in $\mathcal{I}$ satisfies all answers thus far. In other words, the cardinality of the *version space* U_ℓ at time ℓ:

$$U_\ell = \{i \in \mathcal{I} : \bigwedge_{j=1}^{\ell-1} (q_j(i) = q_j(i^*))\} \tag{2}$$

is one. We let:

$$N(T, i) = \arg\min_{\ell} [U_\ell = \{i^*\} \mid T, i^* = i] \tag{3}$$

be the number of questions after which the item i is uniquely identified by the policy T.

Interaction with the user is usually costly. As a result, good content discovery systems should be able to find target items in as few interactions as possible. In this work, we focus on learning systems that minimize the *expected number of interactions* with the user:

$$T^* = \arg\min_{T} \mathbb{E}_{i \sim \pi^*}[N(T, i)] \tag{4}$$

given user preferences π^*.

Consider the case where we want to determine the best decision tree T given prior knowledge of the user preferences π^*. This case is well studied in stochastic control, where it can be formulated as a Markov decision process (MDP) and solved by dynamic programming (Bertsekas, 2012). While dynamic programming can be used to learn the optimal policy T^*, it is computationally intractable for large question pools due to the *curse of dimensionality* (Bertsekas, 2012). If the system is allowed to ask arbitrary questions, not confined to a specific question pool $\mathcal{Q}$, then the optimal policy T^* can be found efficiently by Huffman coding (Cover & Thomas, 2006). Note that this approach cannot be applied to our problem due to the restricted question set.

3. Related Work

Our content discovery problem is an instance of *generalized binary search (GBS)* (Dasgupta, 2005; Nowak, 2011). In GBS, the goal is to learn a policy that finds items in a collection of items in the minimum number

Algorithm 1 GBS($\mathcal{I}, \mathcal{Q}, \pi^*$)
Greedy generalized binary search.

Input: Items $\mathcal{I}$, questions $\mathcal{Q}$, and user preferences π^*

$U_1 \leftarrow \mathcal{I}$
$\ell \leftarrow 1$
while $|U_\ell| > 1$ **do**
  Split the version space U_ℓ closest to two halves:
  $$q_\ell = \arg\min_q \left| \sum_{i \in U_\ell} (2\mathbb{1}\{q(i) = 1\} - 1)\pi^*(i) \right|$$
  Ask the question q_ℓ and observe the reply $q_\ell(i^*)$
  Reduce U_ℓ based on the answer:
  $$U_{\ell+1} = U_\ell \cap \{i \in \mathcal{I} : q_\ell(i) = q_\ell(i^*)\}$$
  $\ell \leftarrow \ell + 1$
end while

Output: Target item $i^* \in U_\ell$

of queries. The optimal policy is NP-hard to compute, both in the worst and average cases. A nearly-optimal policy can be computed greedily by an algorithm that always selects the query that divides the version space closest to two halves. We refer to this solution as GBS (Algorithm 1).

The number of queries in Algorithm GBS was bounded both in the worst (Dasgupta et al., 2003) and average (Dasgupta, 2005) cases. The average-case analysis was later improved (Golovin & Krause, 2011) based on the notion of adaptive submodularity. The resulting upper bound is presented below.

Lemma 1. *[Theorem 5.8 (Golovin & Krause, 2011)] Let π^* be a probability distribution over items. Let T^* be the optimal policy with respect to π^* and T^{g} be the greedy policy generated by Algorithm* GBS*. Then:*

$$\mathbb{E}_{i \sim \pi^*}[N(T^{\mathrm{g}}, i)] \leq (1 - \ln(\pi^*_{\min}))\mathbb{E}_{i \sim \pi^*}[N(T^*, i)],$$

where $\pi^*_{\min} = \min_{i \in \mathcal{I}} \pi^*(i)$.

In practice, the distribution π^* is usually unknown. In our paper, we study the problem where π^* is revealed sequentially through interactions with the user. This is the first work that derives performance guarantees for this setting and shows how the quality of GBS policies improves over time with more interactions.

Yue & Guestrin (2011) show how to learn policies that cover items of user's interest based on interaction with the user. Our paper is related since both problems are submodular and involve learning of user preferences. Our work differs from Yue & Guestrin (2011) in several aspects. First, we study a variant of the minimum set cover problem. The problem in Yue & Guestrin (2011) is a maximum coverage problem. Second, we guide the user to the items of interest by questions, rather than recommending k items at once. So naturally, both our solution and its analysis differ significantly from those of Yue & Guestrin (2011).

The problem of learning trees in the online setting has been studied before, for instance in the framework of experts (Section 5.3 in Cesa-Bianchi & Lugosi (2006)). In this setting, the best tree can be found by learning $|Q|(2^{D+1} - 1)$ weights, where $|Q|$ is the number of the experts at each node of the tree and D is the depth of the tree. In our setting (Section 2), the trees can be as deep as $\min\{M, |Q|\}$. Therefore, online learning with tree experts would be exponential in the quantities of interest, $\min\{M, |Q|\}$, and impractical for solving our problem.

4. Sequential Bayesian Search

In most real-world problems, the user preferences π^* are initially unknown. By repeatedly interacting with the user, we can learn the preferences and reduce the number of asked questions to find target items. In this section, we propose a novel algorithm for interactive search that learns user preferences. We refer to it as *Sequential Bayesian Search* (SBS).

We employ Bayesian learning for two reasons. First, note that the choice of the target item is not affected by how we ask questions (Section 2). Therefore, we do not need to *explore*, which implies that learning and optimization can be separated. Second, high-quality prior knowledge can significantly reduce the expected number of interactions with the user in practice. So Bayesian learning seems like an appropriate approach to our problem.

Specifically, we assume that the system has a prior belief $\mathbb{P}_0$ over the user preferences π^*, which is a probability density function over all the possible realizations of π^*. The system updates its belief at the end of each episode after observing the target item in that episode, which is sampled i.i.d. from π^*. During episode t, the system uses its current belief $\mathbb{P}_t$ to derive a policy that minimizes the expected number of questions to find the target item:

$$T^*_t = \min_T \mathbb{E}_{\pi \sim \mathbb{P}_t} \left[\mathbb{E}_{i \sim \pi} \left[N(T, i) \right] \right]. \tag{5}$$

In summary, we find the best policy T^*_t with respect to the system's belief $\mathbb{P}_t$. The optimization problem in Equation 5 can be simplified based on the notion of the certainty-equivalent user preference π^*_t.

Definition 1. *Let $\mathbb{P}_t$ be the system's belief over π^* in episode t. Then the* ***certainty-equivalent (CE)*** *user preference π^* is defined as $\pi^*_t(i) = \mathbb{E}_{\pi \sim \mathbb{P}_t}[\pi(i)]$ for all $i \in \mathcal{I}$.*

Algorithm 2 $\texttt{SBS}^*(\mathcal{I}, \mathcal{Q}, \mathbb{P}_0)$
Optimal sequential Bayesian search.

Input: Items $\mathcal{I}$, questions $\mathcal{Q}$, and prior belief $\mathbb{P}_0$ on user preferences

for episode $t = 0, 1, 2, \ldots$ **do**
  Compute the CE user preference:
    $\pi_t^*(i) = \mathbb{E}_{\pi \sim \mathbb{P}_t}[\pi(i)] \quad \forall i \in \mathcal{I}$
  Let the user find the target item i_t:
    Solve $T_t^* = \arg\min_T \mathbb{E}_{i \sim \pi_t^*}[N(T, i)]$
    Apply T_t^* to obtain i_t
  Update the system's belief $\mathbb{P}_t$ using Bayes' rule:
    $\mathbb{P}_{t+1}(\pi) \propto \pi(i_t)\mathbb{P}_t(\pi) \quad \forall \pi$
end for

Based on this definition, we make the following observation. Computing the optimal policy with respect to belief $\mathbb{P}_t$ is equivalent to computing the optimal policy with respect to the CE user preference π_t^*. This observation is formalized in Lemma 2. We also note that $\min_T \mathbb{E}_{i \sim \pi_t^*}[N(T, i)]$ is computationally equivalent to finding the optimal tree, where π^* is replaced by π_t^*.

Lemma 2. *The problem* $\min_T \mathbb{E}_{\pi \sim \mathbb{P}_t}[\mathbb{E}_{i \sim \pi}[N(T, i)]]$ *is equivalent to* $\min_T \mathbb{E}_{i \sim \pi_t^*}[N(T, i)]$ *in all episodes* t.

Proof. For any policy T:

$$\mathbb{E}_{\pi \sim \mathbb{P}_t}[\mathbb{E}_{i \sim \pi}[N(T, i)]] = \mathbb{E}_{\pi \sim \mathbb{P}_t}\left[\sum_{i \in \mathcal{I}} \pi(i) N(T, i)\right] =$$
$$\sum_{i \in \mathcal{I}} N(T, i)\mathbb{E}_{\pi \sim \mathbb{P}_t}[\pi(i)] = \sum_{i \in \mathcal{I}} \pi_t^*(i) N(T, i) =$$
$$\mathbb{E}_{i \sim \pi_t^*}[N(T, i)].$$

□

4.1. Algorithm $\texttt{SBS}^*$

In this section, we propose a novel content discovery algorithm $\texttt{SBS}^*$ (Algorithm 2). In each episode t, the algorithm involves three steps: (1) estimating the CE user preference π_t^*, (2) solving the optimization problem $T_t^* = \arg\min_T \mathbb{E}_{i \sim \pi_t^*}[N(T, i)]$, and (3) updating the belief $\mathbb{P}_t$ based on Bayes' rule. Due to Lemma 2, $\min_T \mathbb{E}_{i \sim \pi_t^*}[N(T, i)]$ is computationally equivalent to finding the optimal tree. This problem can be solved efficiently when the question pool $\mathcal{Q}$ is small, or if the system can ask arbitrary questions (Section 2).

Note that the user preferences π^* are a multinomial distribution over items $\mathcal{I}$. So the computation of the CE user preference π_t^*, and updating the belief over π^*, can be implemented efficiently when the prior $\mathbb{P}_0$ is the Dirichlet distribution, the conjugate prior of the multinomial distribution. Specifically, we assume that the system's prior belief is modeled as a Dirichlet distribution with parameters $\alpha \in \Re_+^M$, denoted as $\mathrm{Dir}(\alpha)$, and we define an indicator vector $Z_t \in \Re^M$ such that $Z_t(i) = \mathbb{1}\{i = i_t\}$, where i_t represents the target item in episode t.

From Bayes' rule, the posterior belief at the beginning of episode t is:

$$\mathbb{P}_t = \mathrm{Dir}\left(\alpha + \textstyle\sum_{\tau=0}^{t-1} Z_\tau\right). \tag{6}$$

Furthermore, from the properties of the Dirichlet distribution, we have:

$$\pi_t^*(i) = \mathbb{E}_{\pi \sim \mathbb{P}_t}[\pi(i)] = \frac{\alpha(i) + \sum_{\tau=0}^{t-1} Z_\tau(i)}{(\sum_{i' \in \mathcal{I}} \alpha(i')) + t}, \tag{7}$$

where $\alpha(i)$ is the i-th entry of α, which corresponds to item i. So the CE user preference π_t^* can be updated by counting and re-normalization.

By design and Lemma 2, Algorithm $\texttt{SBS}^*$ is Bayesian optimal.

Proposition 1. *Algorithm* $\texttt{SBS}^*$ *is Bayesian optimal. That is, for any episode* t*:*

- $\mathbb{P}_t$ *is the posterior belief over* π^* *at the beginning of episode* t*;*
- $T_t^* = \arg\min_T \mathbb{E}_{\pi \sim \mathbb{P}_t}[\mathbb{E}_{i \sim \pi}[N(T, i)]]$.

Algorithm 3 $\texttt{SBS}(\mathcal{I}, \mathcal{Q}, \mathbb{P}_0)$
Greedy sequential Bayesian search.

Input: Items $\mathcal{I}$, questions $\mathcal{Q}$, and prior belief $\mathbb{P}_0$ on user preferences

for episode $t = 0, 1, 2, \ldots$ **do**
  Compute the CE user preference:
    $\pi_t^*(i) = \mathbb{E}_{\pi \sim \mathbb{P}_t}[\pi(i)] \quad \forall i \in \mathcal{I}$
  Let the user find the target item i_t:
    $i_t \leftarrow \texttt{GBS}(\mathcal{I}, \mathcal{Q}, \pi_t^*)$
  Update the system's belief $\mathbb{P}_t$ using Bayes' rule:
    $\mathbb{P}_{t+1}(\pi) \propto \pi(i_t)\mathbb{P}_t(\pi) \quad \forall \pi$
end for

4.2. Algorithm $\texttt{SBS}$

Unfortunately, for a general question set $\mathcal{Q}$, computation of the optimal policy T_t^* is intractable. However, the policy can be approximated efficiently by a greedy algorithm (Section 3). This motivates us to propose a near-optimal algorithm $\texttt{SBS}$ (Algorithm 3) for content discovery.

The policy generated by Algorithm $\texttt{SBS}$ in episode t is denoted by T_t^{g}. Similarly to Algorithm $\texttt{SBS}^*$, the new

method is computationally efficient when its belief $\mathbb{P}_t$ is modeled by a Dirichlet distribution. The following proposition claims that our method is Bayesian near-optimal.

Proposition 2. *Algorithm* SBS *is Bayesian near optimal. That is, for any episode t:*

- *$\mathbb{P}_t$ is the posterior belief over π^* at the beginning of episode t;*
- *If $\min_{i\in\mathcal{I}} \pi_t^*(i) > 0$, then:*[1]

$$\mathbb{E}_{\pi\sim\mathbb{P}_t}[\mathbb{E}_{i\sim\pi}[N(T_t^{\mathrm{g}}, i)]] \leq \left[1 - \ln\left(\min_{i\in\mathcal{I}} \pi_t^*(i)\right)\right] \min_T \mathbb{E}_{\pi\sim\mathbb{P}_t}[\mathbb{E}_{i\sim\pi}[N(T, i)]],$$

where π_t^* is the CE user preference in episode t.

Proof. From Algorithm SBS, $\mathbb{P}_t$ is the posterior belief over π^*. From Lemma 1, we have:

$$\mathbb{E}_{i\sim\pi_t^*}[N(T_t^{\mathrm{g}}, i)] \leq \left[1 - \ln\left(\min_{i\in\mathcal{I}} \pi_t^*(i)\right)\right] \min_T \mathbb{E}_{i\sim\pi_t^*}[N(T, i)].$$

Furthermore, similarly to Lemma 2, we have:

$$\mathbb{E}_{i\sim\pi_t^*}[N(T, i)] = \mathbb{E}_{\pi\sim\mathbb{P}_t}[\mathbb{E}_{i\sim\pi}[N(T, i)]]$$
$$\mathbb{E}_{i\sim\pi_t^*}[N(T_t^{\mathrm{g}}, i)] = \mathbb{E}_{\pi\sim\mathbb{P}_t}[\mathbb{E}_{i\sim\pi}[N(T_t^{\mathrm{g}}, i)]].$$

Our second claim follows directly from combining the above three equations. □

5. Analysis

In Section 4, we showed that Algorithms SBS* and SBS are Bayesian optimal and Bayesian near-optimal, respectively. In other words, both SBS* and SBS achieve good performance with respect to the system's current belief. However, such results do not reflect the cost of learning user preferences π^*. In particular, since SBS* and SBS learn π^* while interacting with the user, they incur a performance loss, ask more questions than the optimal policy T^*. We show that these losses are small by analyzing our algorithms from the frequentist perspective. In particular, we show that the *cumulative regret* of Algorithm SBS* and the $[1 - \ln(\pi^*_{\min})]$-*scaled cumulative regret* of Algorithm SBS are sublinear. Note that from the frequentist perspective, π^* is deterministic but unknown.

We assume that the system's prior belief is a Dirichlet distribution $\mathrm{Dir}(\alpha)$. To simplify notation, we denote the minimum expected number of interactions under the user preferences π^* by $N^* = \min_T \mathbb{E}_{i\sim\pi^*}[N(T, i)]$. Furthermore, we define the minimum user preference probability $\pi^*_{\min} = \min_{i\in\mathcal{I}} \pi^*(i)$ and $\alpha_0 = \sum_{i\in\mathcal{I}} \alpha(i)$, where $\alpha(i)$ is the i-th entry of α.

[1] This condition holds when $\mathbb{P}_0 = \mathrm{Dir}(\alpha)$ and $\alpha(i) > 0$ for all $i \in \mathcal{I}$.

5.1. Analysis of Algorithm SBS*

The performance of Algorithm SBS* is measured by its cumulative regret.

Definition 2. *For any episode τ, the* ***cumulative regret*** *of Algorithm* SBS* *is defined as:*

$$\mathbf{Reg}^*(\tau) = \sum_{t=0}^{\tau} \mathbb{E}_{T_t^*}[\mathbb{E}_{i\sim\pi^*}[N(T_t^*, i)]] - (\tau+1)N^*.$$

The cumulative regret $\mathbf{Reg}^*(\tau)$ is the expected cumulative performance loss of Algorithm SBS* in the first $\tau+1$ episodes, relatively to the optimum $(\tau+1)N^*$. In Theorem 1, we prove that $\mathbf{Reg}^*(\tau)$ is sublinear in τ when τ is sufficiently large.

Theorem 1. *Let $\mathbb{P}_0 \sim \mathrm{Dir}(\alpha)$ and τ_E be the sample threshold of Algorithm* SBS* *defined as:*

$$\tau_E = \min\left\{t \geq 4 : \frac{\ln(t)}{t} \leq \left(\frac{\pi^*_{\min}}{6}\right)^2 \text{ and } \frac{4}{3}\alpha_0 \max_{i\in\mathcal{I}} \left|\frac{\alpha(i)}{\alpha_0} - \pi^*(i)\right| \leq [t\ln(t)]^{\frac{1}{2}}\right\}.$$

Then, for $\tau < \tau_E$, $\mathbf{Reg}^*(\tau) \leq |\mathcal{Q}|(\tau+1)$. In addition, for $\tau \geq \tau_E$:

$$\mathbf{Reg}^*(\tau) \leq |\mathcal{Q}|\tau_E + \frac{2M|\mathcal{Q}|}{\tau_E - 1} + \frac{12N^*}{\pi^*_{\min}} \left\{[\tau\ln(\tau)]^{\frac{1}{2}} - [(\tau_E - 1)\ln(\tau_E - 1)]^{\frac{1}{2}}\right\}.$$

The proof of Theorem 1 is sketched below. The complete proof can be found in Appendix.

Proof. If the CE user preference π_t^* is close enough to the true user preference π^*, the policy T_t^* in episode t performs near optimally. Specifically, if $\|\pi^* - \pi_t^*\|_\infty < \pi^*_{\min}$, then:

$$\mathbb{E}_{i\sim\pi^*}[N(T_t^*, i)] - N^* \leq \frac{2\|\pi^* - \pi_t^*\|_\infty}{\pi^*_{\min} - \|\pi^* - \pi_t^*\|_\infty} N^*.$$

Based on Hoeffding's inequality, $\|\pi_t^* - \pi^*\|_\infty$ is small with high probability for $t \geq \tau_E$. Hence, the expected regret in episode t is small. Specifically, for all $t \geq \tau_E$, we have:

$$\mathbb{E}_{T_t^*}[\mathbb{E}_{i\sim\pi^*}[N(T_t^*, i)]] - N^* < \frac{6}{\pi^*_{\min}} \left[\frac{\ln(t)}{t}\right]^{\frac{1}{2}} N^* + \frac{2M|\mathcal{Q}|}{t^2}.$$

For $t < \tau_E$, the regret can be bounded naively as:

$$\mathbb{E}_{T_t^*}[\mathbb{E}_{i\sim\pi^*}[N(T_t^*, i)]] - N^* \leq |\mathcal{Q}|.$$

Our bound on the cumulative regret up to episode τ is proved by summing up the above upper bounds. □

Theorem 1 states that $\mathbf{Reg}^*(\tau) = O([\tau \ln(\tau)]^{\frac{1}{2}})$ when $\tau \geq \tau_E$. Therefore, τ_E can be viewed as the number of episodes after which Algorithm SBS* achieves good performance. Note that τ_E depends on both $\pi^*_{\min}$ and the choice of $\mathbb{P}_0$. In particular, smaller $\pi^*_{\min}$ or a poor choice of $\mathbb{P}_0$ result in larger τ_E.

5.2. Analysis of Algorithm SBS

Note that the cumulative regret (Definition 2) is not a good metric for evaluating the performance of Algorithm SBS. Specifically, since the algorithm computes a suboptimal solution, it can incur a large loss in comparison to N^* in each episode t and this loss does not diminish as $t \to \infty$. Lemma 1 suggests that the algorithm should be compared to the $[1 - \ln(\pi^*_{\min})]$ factor of the minimum expected number of interactions N^*. This motivates us to generalize the notion of the cumulative regret to a C-scaled cumulative regret.

Definition 3. *For any episode τ and $C \geq 1$, the **C-scaled cumulative regret** of Algorithm SBS is defined as:*

$$\mathbf{Reg}(\tau, C) = \sum_{t=0}^{\tau} \mathbb{E}_{T_t^{\mathrm{g}}}[\mathbb{E}_{i\sim\pi^*}[N(T_t^{\mathrm{g}}, i)]] - (\tau + 1)CN^*.$$

The regret $\mathbf{Reg}(\tau, C)$ measures the expected cumulative loss of Algorithm SBS in the first $\tau + 1$ episodes, relatively to the C-scaled optimum $(\tau + 1)CN^*$. Note that $C = 1$ yields the classical cumulative regret. We would like to bound $\mathbf{Reg}(\tau, 1 - \ln(\pi^*_{\min}))$ from above. In Theorem 2, we prove that $\mathbf{Reg}(\tau, 1 - \ln(\pi^*_{\min}))$ is sublinear in τ when τ is sufficiently large.

Theorem 2. *Let $\mathbb{P}_0 \sim \mathrm{Dir}(\alpha)$ and τ_G be the sample threshold of Algorithm SBS defined as:*

$$\tau_G = \min \left\{ t \geq 4 : \frac{\ln(t)}{t} \leq \left(\frac{0.0689\pi^*_{\min}}{\ln\left(\frac{e}{\pi^*_{\min}}\right)} \right)^2 \text{ and } \frac{4}{3}\alpha_0 \max_{i\in\mathcal{I}} \left| \frac{\alpha(i)}{\alpha_0} - \pi^*(i) \right| \leq [t \ln(t)]^{\frac{1}{2}} \right\}.$$

Then, for $\tau < \tau_G$, $\mathbf{Reg}(\tau, 1 - \ln(\pi^*_{\min})) \leq |\mathcal{Q}|(\tau + 1)$. Moreover, for $\tau \geq \tau_G$:

$$\mathbf{Reg}(\tau, 1 - \ln(\pi^*_{\min})) \leq |\mathcal{Q}|\tau_G + \frac{2M|\mathcal{Q}|}{\tau_G - 1} + f(\pi^*_{min}, N^*, \tau, \tau_G),$$

where $f(\pi^*_{min}, N^*, \tau, \tau_G) =$

$$\frac{40 \ln\left(\frac{e}{\pi^*_{\min}}\right) N^*}{\pi^*_{\min}} \left([\tau \ln(\tau)]^{\frac{1}{2}} - [(\tau_G - 1)\ln(\tau_G - 1)]^{\frac{1}{2}} \right).$$

The proof of Theorem 2 is sketched below. The complete proof can be found in Appendix.

Proof. If the CE user preference π^*_t is close enough to the true user preference π^*, the policy T_t^{g} in episode t performs near optimally. Specifically, if $\|\pi^* - \pi^*_t\|_\infty < \pi^*_{\min}$, then:

$$\mathbb{E}_{i\sim\pi^*}[N(T_t^{\mathrm{g}}, i)] \leq \left[\frac{\pi^*_{\min} + \|\pi^* - \pi^*_t\|_\infty}{\pi^*_{\min} - \|\pi^* - \pi^*_t\|_\infty} \right]^2 \ln\left(\frac{e}{\pi^*_{\min} - \|\pi^* - \pi^*_t\|_\infty} \right) N^*.$$

Based on Hoeffding's inequality, $\|\pi^*_t - \pi^*\|_\infty$ is small with high probability for $t \geq \tau_G$. Hence, the expected scaled regret in episode t is small. Specifically, for all $t \geq \tau_G$, we have:

$$\mathbb{E}_{T_t^{\mathrm{g}}}[\mathbb{E}_{i\sim\pi^*}[N(T_t^{\mathrm{g}}, i)]] - \ln\left(\frac{e}{\pi^*_{\min}}\right) N^* < \left[8 + 12 \ln\left(\frac{e}{\pi^*_{\min}}\right) \right] \frac{1}{\pi^*_{\min}} \left[\frac{\ln(t)}{t} \right]^{\frac{1}{2}} N^* + \frac{2M|\mathcal{Q}|}{t^2}.$$

For $t < \tau_G$, the regret can be bounded naively as:

$$\mathbb{E}_{T_t^{\mathrm{g}}}[\mathbb{E}_{i\sim\pi^*}[N(T_t^{\mathrm{g}}, i)]] - \ln\left(\frac{e}{\pi^*_{\min}}\right) N^* \leq |\mathcal{Q}|.$$

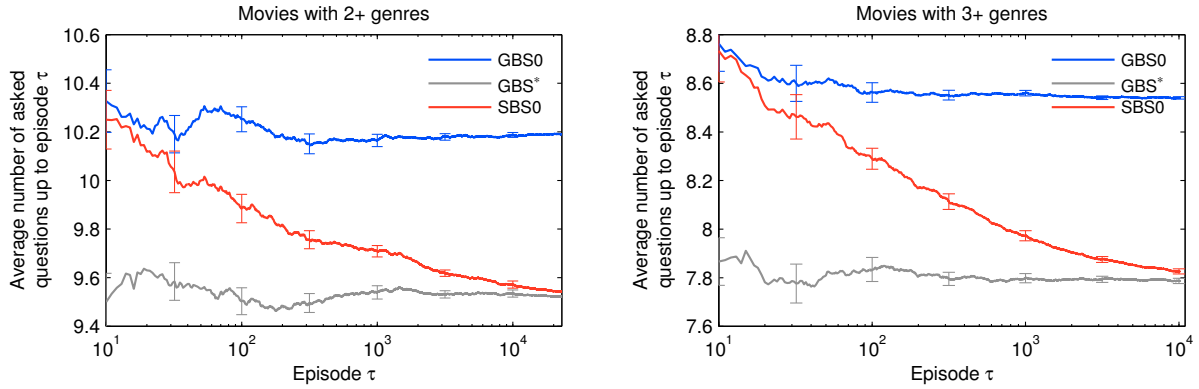
Our bound on the cumulative regret up to episode τ is proved by summing up the above upper bounds. □

The above theorem says that $\mathbf{Reg}(\tau, 1 - \ln(\pi^*_{\min})) = O([\tau \ln(\tau)]^{\frac{1}{2}})$ when $\tau \geq \tau_G$. The value of τ_G depends on $\pi^*_{\min}$ and the choice of $\mathbb{P}_0$. Specifically, smaller $\pi^*_{\min}$ or a poor choice of $\mathbb{P}_0$ result in larger τ_G.

Note that the $\frac{1}{\pi^*_{\min}}$ factor in the regret bounds in Theorems 1 and 2 originates in Lemma A-1 in Appendix and is due to bounding the terms that grow with time τ using N^*, as opposing to looser $|\mathcal{Q}|$. This factor is hard to eliminate because the bound in Lemma A-1 is tight.

6. Experiments

Algorithm SBS is evaluated on a real-world movie discovery problem. We perform two experiments. First, we study how the number of questions asked by SBS decreases over time, and compare the approach to two baselines. Second, we show how the choice of the prior $\mathbb{P}_0$ affects the quality of SBS policies.



(a) (b)

Figure 1. Comparison of three movie discovery policies on *MovieLens* dataset. The methods are compared by the average number of questions asked up to episode τ.

Movie genres		
Action	Adventure	Animation
Children's	Comedy	Crime
Documentary	Drama	Fantasy
Film-Noir	Horror	Musical
Mystery	Romance	Sci-Fi
Thriller	War	Western

Table 1. Movie genres in *MovieLens* dataset.

6.1. Experimental Setup

Our experiment is conducted on a dataset of one million ratings from the *MovieLens* project (Lam & Herlocker, 2012). In this dataset, 6k users rate 4k movies for more than one year. The movies are annotated by 19 genres (Table 1). Only 302 unique combinations of genres exist in our dataset. In our experiments, these are the individual items in $\mathcal{I}$. Movies with the same genre descriptors are equivalent in the sense that the user has equal preferences for all of these movies.

We assume that each rating event is a movie viewing event, where Algorithm SBS is used to find the target movie, the movie that was actually watched. We ask questions of the form "Would you like to watch X?", where X represents a genre, and measure the number of asked questions until the target movie is identified. We learn a single preference distribution for all users. We opt for this setup since we want to show that SBS converges to the optimal solution when the number of episodes is large. None of our individual users rated enough movies to illustrate this trend.

We focus on two subsets of our data, movies belonging to at least two, and at least three, genres. The movies in the former set are harder to identify since they are described by fewer attributes. These movies are rated over 700k times, and the number of unique movies is $M = 283$. The movies in the latter set are rated 300k times, and the number of unique movies is $M = 203$. To avoid bias, we partition both datasets into 30 time periods with the same number of ratings, evaluate the policy in each time period, and finally average the results over the periods.

6.2. Optimality

In the first experiment, we study how SBS policies improve over time. Algorithm SBS is initialized with an uninformative prior $\mathbb{P}_0 = \mathrm{Dir}(\mathbf{1})$, where $\mathbf{1}$ is a vector of all ones. We refer to this policy as SBS0.

Our approach is compared to two baselines. The first baseline GBS0 assumes that all items are drawn with the same probability, $\mathtt{GBS}(\mathcal{I}, \mathcal{Q}, \pi_0)$ where $\pi_0(i) = \frac{1}{M}$. This can be viewed as an upper bound on the number of questions asked by SBS0. The second baseline GBS* knows the probability π^* with which target items are drawn, $\mathtt{GBS}(\mathcal{I}, \mathcal{Q}, \pi^*)$. This can be viewed as a lower bound on the number of questions asked by SBS0. The performance of all policies is measured by the average number of questions asked up to episode τ. The lower the number, the better.

Our results are reported in Figure 1. In Figure 1a, the average number of questions asked by SBS0 decreases over time to 9.5. This is 0.7 questions less than GBS0, which asks 10.2 questions, and similar to the best solution in hindsight GBS*. Note that the Shannon entropy $H(\pi^*)$ of π^* is a lower bound on the expected number of questions to find a randomly chosen item from π^*. In our case, $H(\pi^*) \approx 6.8$. Relatively to this baseline, the improvement from 10.2 to 9.5 is 21%.

Similar trends can be observed in Figure 1b. The average number of questions asked by SBS0 decreases over time to 7.8. This is 0.7 questions less than GBS0, which

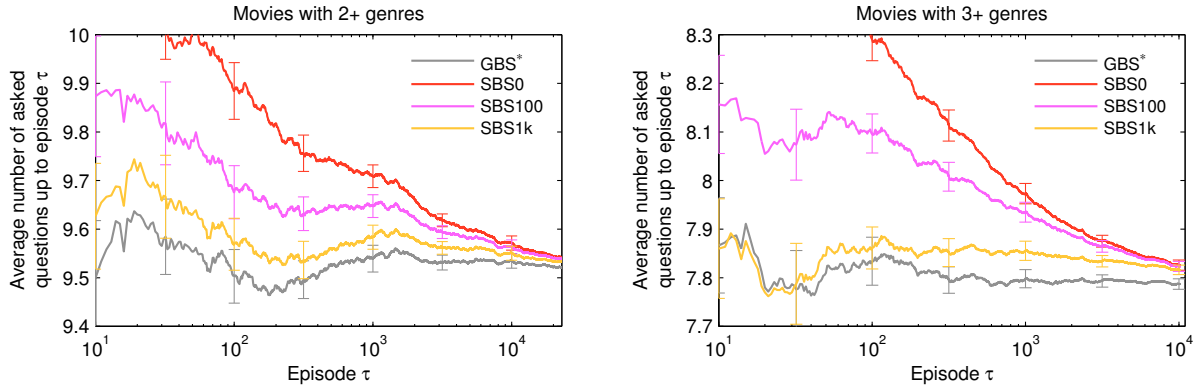


Figure 2. Comparison of four movie discovery policies on *MovieLens* dataset. The methods are compared by the average number of questions asked up to episode τ.

asks 8.5 questions, and similar to the best solution in hindsight GBS*. Relatively to $H(\pi^*)$, the improvement from 8.5 to 7.8 is 41%.

6.3. Impact of the Prior $\mathbb{P}_0$

In the second experiment, we study how a well-chosen prior $\mathbb{P}_0$ helps in speeding up the convergence of SBS to the lower bound GBS*. We compare three variants of SBS. In the first variant SBS0, the prior $\mathbb{P}_0 = \mathrm{Dir}(\mathbf{1})$ is uninformative. In the second and third variants, the prior is $\mathbb{P}_0 = \mathrm{Dir}(\mathbf{1} + c)$, where $c \in \Re^M$ and $c(i)$ is the number of occurrences of item i in 100 and 1k random movie viewing events, respectively. We refer to these variants as SBS100 and SBS1k, respectively.

Our results are summarized in Figure 2. In both plots, we observe that SBS1k asks less questions on average than SBS100, which asks less questions than SBS0. In other words, SBS performs better when the prior $\mathbb{P}_0$ is more accurate. All variants of SBS approach the lower bound GBS* as time increases.

7. Conclusions and Discussion

In this work, we propose two user-adaptive algorithms for interactive content discovery, SBS* and SBS. Both algorithms learn user preferences over time as the user interacts with them. We proved that SBS* is Bayesian optimal and achieves sublinear cumulative regret; and that SBS is Bayesian near-optimal and achieves sublinear $[1 - \ln(\pi^*_{\min})]$-scaled cumulative regret. We evaluate SBS on a real-world problem and demonstrate that its performance is similar to the best solution in hindsight.

We assume that user's target items i^* are drawn i.i.d. from a stationary distribution π^* (Section 2) to simplify the exposition of our work. This assumption can be quite easily relaxed. In particular, both of our algorithms can be adapted to the target items that are drawn from time-varying distributions or follow an exogenous Markov chain. As long as the user preferences are exogenous, our algorithms would work. The only difference would be that learning of more complicated models takes more time.

This is the first work that studies learning in adaptive submodularity (Golovin & Krause, 2011). In particular, we show that a nearly optimal policy for solving an adaptive submodular problem whose model is initially unknown, identifying $M-1$ non-target items by asking the minimum number of questions on average, can be learned from solving the problem repeatedly. A key observation in our solution is that the expected gain of asking questions, conditioned on any sequence of questions and answers, can be estimated efficiently from another statistic, the probability of drawing target items π^*. Therefore, our learning problem reduces to estimating π^*. We believe that this decomposition could be useful in solving other adaptive submodular problems.

Perhaps surprisingly, it can be shown that the cumulative regret of Algorithm SBS* is $O(1)$. In particular, learning of the minimum-depth tree can be viewed as choosing one tree out of $|\mathcal{Q}|^M M!$ trees of at most $|\mathcal{Q}|$ depth. Based on the union bound and Hoeffding's inequality, the regret of the policy that chooses in each episode the minimum-depth tree given all past target items is $\Omega(N|\mathcal{Q}|^3)$. This regret bound is problematic since $|\mathcal{Q}|$ is often large. We believe that our bound is much more practical because it is linear in M and $|\mathcal{Q}|$, and deriving such a bound is challenging. Even when t is relatively large, our bound is expected to be tighter than a higher-order polynomial $O(1)$ bound. We leave derivation of practical $O(1)$ bounds for future work.

References

Bertsekas, Dimitri. *Dynamic Programming and Optimal Control*. Athena Scientific, Nashua, New Hampshire, 2012.

Cesa-Bianchi, Nicolo and Lugosi, Gabor. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, 2006.

Cover, Thomas and Thomas, Joy. *Elements of Information Theory*. John Wiley & Sons, Hoboken, New Jersey, 2006.

Dasgupta, Sanjoy. Analysis of a greedy active learning strategy. In *Advances in Neural Information Processing Systems 17*, pp. 337–344, 2005.

Dasgupta, Sanjoy, Lee, Wee Sun, and Long, Philip. A theoretical analysis of query selection for collaborative filtering. *Machine Learning*, 51(3):283–298, 2003.

Golovin, Daniel and Krause, Andreas. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

Lam, Shyong and Herlocker, Jon. MovieLens 1M Dataset. http://www.grouplens.org/node/12, 2012.

Nowak, Robert. The geometry of generalized binary search. *IEEE Transactions on Information Theory*, 57(12):7893–7906, 2011.

Yue, Yisong and Guestrin, Carlos. Linear submodular bandits and their application to diversified retrieval. In *Advances in Neural Information Processing Systems 24*, pp. 2483–2491, 2011.