Learning to Discover: A Bayesian Approach

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

Branislav Kveton and **Sandilya Bhamidipati**
Technicolor Labs
Palo Alto, CA
branislav.kveton@technicolor.com
sandilya.bhamidipati@technicolor.com

Abstract

Generalized binary search is a natural framework for modeling interactive search. This is the first paper that studies this problem under the assumption that the user's searched items are drawn from an unknown probability distribution. We propose an algorithm that efficiently learns how to quickly discover the user's searched items over time as the user interacts with the search engine, show that it is Bayesian optimal, and prove that its regret increases only sublinearly with time.

1 Introduction

Extensive online collections of content exist, such as restaurants, movies, music, and others. Items in these collections are typically described by many attributes, some of which are carefully curated using labels that are meaningful and common, and others that are freely generated by users, such as reviews. In this paper, we address the problem of content discovery in this setting.

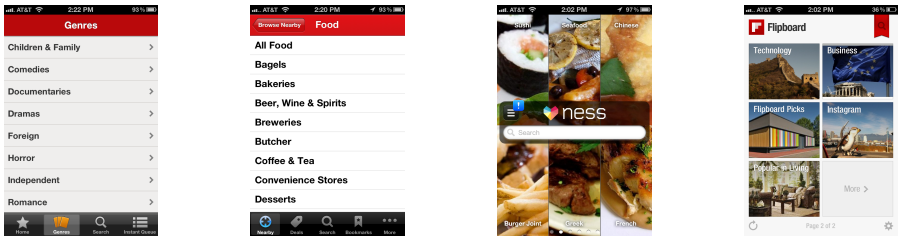
The problem of content discovery has been traditionally approached in two ways. In one approach, the user types attributes of interest into in a search box and the system (search engine) returns a ranked list of items that are most relevant to the user's query. In the other, the user navigates through a series of menus with item categories and progressively refines the selection to a list of desired items (Figure 1). Most recommendation websites, like Netflix and Yelp, combine both approaches. In this paper, we focus on the latter approach.

The number of items is typically large. Therefore, they are organized in a tree-like taxonomy, where each node corresponds to a selection in a menu done by the user. Unfortunately, the taxonomic tree is often unbalanced and as a result not all items can be found in a small number of menu selections. On the other hand, the branching of the tree is sometimes too large, which may impact the usability of the system. For example, Yelp users can choose from over 100 cuisines but only 9 can be shown simultaneously on the screen of a smart phone (Figure 1b). Finally, in many interfaces, like Netflix (Figure 1a), the taxonomic tree is shallow, only two or three levels deep, and the user is confronted with movies that satisfy the criteria but also include many unwanted items.

In this paper, we view the content discovery problem as an interactive game. In this game, the user has a specific target item on their mind, which is drawn i.i.d. from some probability distribution π^*. In each turn of the game, the system asks a (binary) question, the user answers, and this process is repeated until the system identifies a single item that satisfies all user's answers so far. Our goal is to minimize the expected number of asked questions. This problem is known as *generalized binary search (GBS)* [2, 5] and can be solved greedily nearly optimally. The greedy solution always selects the question that divides the version space of items closest to two halves.

Golovin and Krause [3] recently showed that GBS is an adaptively submodular problem and further tightened the bound on the number of asked questions. Jedynak *et al.* [4] studied GBS with noisy answers, and showed that when the version space is continuous the greedy algorithm minimizes the entropy of the posterior.

The probability π^* can be viewed as user's preferences and typically it is unknown. In this paper, we study how a system (search engine) should learn to make optimal (or near-optimal) GBS searches in a sequence of games without knowing π^*. We propose an efficient algorithm that learns preferences of the user π^* for target items as the user interacts with the system, show that it is Bayesian optimal, and prove that its regret increases only sublinearly with time.



(a) Netflix. (b) Yelp. (c) Likeness. (d) Apple's Flipboard.

Figure 1: Examples of content discovery applications on a cell phone.

2 Model

In this section, we propose a mathematical model for the interactive game. We assume the system has a collection of M items, and each item $m = 1, 2, \cdots, M$ is identified by a feature vector $\phi(m) \in \{0,1\}^K$. We define the item feature space Φ as

$$\Phi = \{\phi : \exists m = 1, 2, \cdots, M \text{ s.t. } \phi = \phi(m)\}.$$

We use $|\Phi|$ to denote the cardinality of Φ. The user's preference is modeled as a probability distribution π^* over Φ. That is, each time when the user interacts with the system (henceforth referred to as a user-system interaction), with probability $\pi^*(\phi)$, he is looking for the item with feature ϕ (we henceforth say that the user's target item feature at that interaction is ϕ). We further assume that the user's preference π^* is time-invariant, and his target item features in different interactions are statistically independent. That is, the target item features are drawn i.i.d. from π^*.

We also assume that during each user-system interaction, the system is allowed to ask a sequence of yes/no (binary) questions from a question pool $\mathcal{Q}$. Each question in $\mathcal{Q}$ is represented as a function $q : \{0,1\}^K \to \{\mathrm{Y}, \mathrm{N}\}$, where "Y" denotes the answer "yes" while "N" denotes the answer "no". Specifically, if the user's target item feature is $\phi \in \Phi$ and the system chooses to ask him question q, then his answer will be $q(\phi) \in \{\mathrm{Y}, \mathrm{N}\}$. In this paper, the question pool $\mathcal{Q}$ can be arbitrary set of such q's, as long as it satisfies the following assumption: whatever the user's target item feature $\phi \in \Phi$ is, if the system has asked him all the questions in $\mathcal{Q}$, then it is able to discover ϕ. We use $|\mathcal{Q}|$ to denote the number of questions in the question pool $\mathcal{Q}$.

During each interaction, the system asks questions adaptively and continues asking questions until it can determine the user's target item. Specifically, assume the user's target item feature is ϕ and the system has asked question $q_1, \cdots, q_n \in \mathcal{Q}$, then the system will observe answers $q_1(\phi), \cdots, q_n(\phi) \in \{\mathrm{Y}, \mathrm{N}\}$. We define the current observation state as $x = (q_i, q_i(\phi))_{i=1}^n$. Furthermore, we say x is a terminal state if the system is able to determine ϕ based on it; otherwise, we say it is a non-terminal state and the system needs to choose a new question to ask based on x. A (one-interaction) policy T is a decision rule to choose a new question given a non-terminal state x (we use notation T since this policy can be represented as a decision tree). Obviously, if both the policy T and the user's target item feature ϕ are given, the sequence of questions the system will ask is completely determined, and we use $N(T, \phi)$ to denote the length of this question sequence.

Generally speaking, the system's objective during each interaction is to minimize the expected number of questions it needs to ask to discover the user's target item. For the oracle case when the system is assumed to know the user preference π^*, in each interaction, it aims to solve

$$\min_T \mathbb{E}_{\phi \sim \pi^*} [N(T, \phi)], \tag{1}$$

which is the well-studied generalized binary search problem. Various methods have been proposed for problem (1). Exact solutions methods include dynamic programming (DP) and Huffman coding. However, the DP method tends to be computationally intractable due to *the curse of dimensionality*, while the Huffman coding method only works for a special case. Approximation algorithms aiming for near-optimal solutions have also been proposed. For example, it is well known that the oracle case of this problem can be solved nearly optimally and computationally efficiently by a greedy algorithm [2, 3]. Recently, based on the notion of adaptive submodularity, the performance bound on this greedy algorithm is further tightened [3].

Of course, in practice, the system does not know π^* beforehand and needs to learn it while interacting with the user. We will discuss this case in detail in Section 3 and 4.

3 Bayesian learning algorithm

As we have discussed in Section 2, in practice, the system needs to learn the user preference π^* while interacting with him. Since we assume that the system cannot influence the user's behavior

Algorithm 1 Bayesian Content Discovery Algorithm

Input: Item feature space Φ, question pool $\mathcal{Q}$ and prior belief $\mathbb{P}_0$

for $t = 0, 1, 2, \cdots$ **do**

Compute the CE user preference $\pi_t^*(\phi) = \mathbb{E}_{\pi\sim\mathbb{P}_t}[\pi(\phi)]$, $\forall \phi \in \Phi$

Solve $\min_T \mathbb{E}_{\phi\sim\pi_t^*}[N(T,\phi)]$ (exactly or approximately), and denote the solution as T_t^*

Apply T_t^* in interaction t, and observe ϕ_t, the user's target item feature in interaction t

Update the system's (posterior) belief based on Bayes rule

$$\mathbb{P}_{t+1}(\pi \in d\pi) = \frac{\mathbb{P}_t(\pi \in d\pi)\,\pi(\phi_t)}{\pi_t^*(\phi_t)}.$$

end for

(the user's target item feature are drawn i.i.d. from π^*, no matter what the system has done), there is no need and no way for the system to *explore*. This distinguishes this problem from the classical multi-armed bandit problems [1] and motivates us to propose a content discovery algorithm based on Bayesian learning, which we refer to as the Bayesian content discovery algorithm.

Specifically, we assume that the system has a prior belief $\mathbb{P}_0$ over the user preference π^*, which is a probability distribution over all the possible realizations of π^*. The system updates its belief at the end of each interaction, after observing the user's target item feature in that interaction (which is an i.i.d. sample from π^*). During interaction t, the system exploits its current belief $\mathbb{P}_t$ to derive an adaptive (one-interaction) policy T_t^*, which (exactly or approximately) solves

$$\min_T \mathbb{E}_{\pi\sim\mathbb{P}_t}\left[\mathbb{E}_{\phi\sim\pi}[N(T,\phi)]\right], \tag{2}$$

and applies it in this interaction. In other words, T_t^* is optimal (or near-optimal) with respect to the system's current belief $\mathbb{P}_t$.

We now show that the optimization problem (2) can be further simplified. Specifically, if we define the certainty-equivalence (CE) user preference in interaction t as $\pi_t^*(\phi) = \mathbb{E}_{\pi\sim\mathbb{P}_t}[\pi(\phi)]$, $\forall \phi \in \Phi$, then for any policy T, we have

$$\begin{aligned}\mathbb{E}_{\pi\sim\mathbb{P}_t}\left[\mathbb{E}_{\phi\sim\pi}[N(T,\phi)]\right] &= \mathbb{E}_{\pi\sim\mathbb{P}_t}[\sum_{\phi\in\Phi}\pi(\phi)N(T,\phi)] = \sum_{\phi\in\Phi}\mathbb{E}_{\pi\sim\mathbb{P}_t}[\pi(\phi)N(T,\phi)] \\ &= \sum_{\phi\in\Phi}N(T,\phi)\mathbb{E}_{\pi\sim\mathbb{P}_t}[\pi(\phi)] = \sum_{\phi\in\Phi}\pi_t^*(\phi)N(T,\phi) = \mathbb{E}_{\phi\sim\pi_t^*}[N(T,\phi)].\end{aligned}$$

Thus, $\min_T \mathbb{E}_{\pi\sim\mathbb{P}_t}[\mathbb{E}_{\phi\sim\pi}[N(T,\phi)]]$ is equivalent to $\min_T \mathbb{E}_{\phi\sim\pi_t^*}[N(T,\phi)]$. Furthermore, notice that $\min_T \mathbb{E}_{\phi\sim\pi_t^*}[N(T,\phi)]$ is mathematically equivalent to the oracle problem (with the true user preference π^* replaced by the CE user preference π_t^*), thus, any method (either exact or approximate) discussed in Section 2 can be used to solve $\min_T \mathbb{E}_{\phi\sim\pi_t^*}[N(T,\phi)]$. The Bayesian content discovery algorithm is detailed in Algorithm 1.

4 Performance of Bayesian learning algorithm

In this section, we analyze the performance of Algorithm 1 from two different perspectives: the frequentist perspective and the Bayesian perspective. First, from the Bayesian perspective, based on the discussion in Section 3, we have:

Theorem 1: *During interaction t, if Algorithm 1 solves $\min_T \mathbb{E}_{\phi\sim\pi_t^*}[N(T,\phi)]$ exactly, then we have*

$$T_t^* \in \arg\min_T \mathbb{E}_{\pi\sim\mathbb{P}_t}\left[\mathbb{E}_{\phi\sim\pi}[N(T,\phi)]\right].$$

That is, T_t^* is Bayesian optimal (with respect to the current belief $\mathbb{P}_t$) if Algorithm 1 solves $\min_T \mathbb{E}_{\phi\sim\pi_t^*}[N(T,\phi)]$ exactly.

Second, from the frequentist perspective, we show that Algorithm 1 achieves a sublinear cumulative scaled regret. Notice that from the frequentist perspective, π^* is deterministic but unknown. Let T^* be an optimal policy that solves the oracle problem, i.e. $T^* \in \arg\min_T \mathbb{E}_{\phi\sim\pi^*}[N(T,\phi)]$. Then $\forall \tau = 0, 1, \cdots$ and $\forall C \geq 1$, the cumulative scaled regret $\mathbf{Reg}(\tau, C)$ of Algorithm 1 is defined as

$$\mathbf{Reg}(\tau, C) = \sum_{t=0}^{\tau}\mathbb{E}_{T_t^*}\left\{\mathbb{E}_{\phi\sim\pi^*}[N(T_t^*,\phi)] - C\mathbb{E}_{\phi\sim\pi^*}[N(T^*,\phi)]\right\}. \tag{3}$$

Notice that $C = 1$ corresponds to the classical cumulative regret. $\mathbf{Reg}(\tau, C)$ measures the expected cumulative performance loss of Algorithm 1 in the first $\tau+1$ interactions, compared to the C-scaled optimal performance.

In the remainder of this section, we consider the special case when the system's prior belief is a Dirichlet distribution with parameter $\alpha \in \Re_+^{|\Phi|}$ (henceforth denoted as $\mathrm{Dir}(\alpha)$). Since Dirichlet distribution is the conjugate distribution of the multinomial distribution, then all the posterior beliefs are also Dirichlet distributions. Furthermore, we define $\alpha_0 = \sum \alpha(\phi)$, where $\alpha(\phi)$ is the component of the vector α corresponding to the item feature ϕ, and $\pi^*_{\min} = \min_{\phi\in\Phi} \pi^*(\phi)$. For any $0 < \delta < \frac{1}{2}$, we define

$$\begin{aligned}
\tau_1(\delta) &= \min\{t > 0 :\ t^{-\frac{1}{2}+\delta} \le \frac{1}{3},\ \frac{4\alpha_0 \max|\frac{\alpha(\phi)}{\alpha_0} - \pi^*(\phi)|}{\pi^*_{\min}} < t^{\frac{1}{2}+\delta} \text{ and } \left[\pi^*_{\min} t^{\delta}\right]^2 \ge 2\ln(t)\}, \qquad (4)\\
\tau_2(\delta) &= \min\{t > 0 :\ t^{-\frac{1}{2}+\delta} \le \frac{1}{32\ln(\frac{e}{\pi^*_{\min}})},\ \frac{4\alpha_0 \max|\frac{\alpha(\phi)}{\alpha_0} - \pi^*(\phi)|}{\pi^*_{\min}} < t^{\frac{1}{2}+\delta} \text{ and } \left[\pi^*_{\min} t^{\delta}\right]^2 \ge 2\ln(t)\},
\end{aligned}$$

where e is the base of natural logarithm. As is shown above, $\tau_1(\delta)$ and $\tau_2(\delta)$ depend on δ, $\pi^*_{\min}$ and the "quality" of $\mathbb{P}_0$. Specifically, smaller δ and $\pi^*_{\min}$ and "worse" $\mathbb{P}_0$ result in larger $\tau_1(\delta)$ and $\tau_2(\delta)$. We have the following theorem on the cumulative scaled regret:

Theorem 2: *Assume* $\mathbb{P}_0 \sim \mathrm{Dir}(\alpha)$*, then* $\forall\tau = 0, 1, \cdots$ *and* $\forall 0 < \delta < \frac{1}{2}$*, with* $\tau_1(\delta)$*,* $\tau_2(\delta)$ *defined in Eqn(4), we have*

$$\mathbf{Reg}(\tau, 1) \le \begin{cases} |\mathcal{Q}|(\tau+1) & \text{if } \tau < \tau_1(\delta) \\ |\mathcal{Q}|\tau_1(\delta) + \frac{6}{1+2\delta}\mathbb{E}_{\phi\sim\pi^*}\left[N(T^*,\phi)\right]\left[\tau^{\frac{1}{2}+\delta} - (\tau_1(\delta)-1)^{\frac{1}{2}+\delta}\right] + |\Phi||\mathcal{Q}|\frac{2}{\tau_1(\delta)-1} & \text{if } \tau \ge \tau_1(\delta) \end{cases},$$

if Algorithm 1 solves $\min_T \mathbb{E}_{\phi\sim\pi^*_t}\left[N(T,\phi)\right]$ *exactly, and* $\mathbf{Reg}\left(\tau, 1-\ln\left(\pi^*_{\min}\right)\right) \le$

$$\begin{cases} |\mathcal{Q}|(\tau+1) & \text{if } \tau < \tau_2(\delta) \\ |\mathcal{Q}|\tau_2(\delta) + \frac{36+32\ln\left(\frac{e}{\pi^*_{\min}}\right)}{1+2\delta}\mathbb{E}_{\phi\sim\pi^*}\left[N(T^*,\phi)\right]\left[\tau^{\frac{1}{2}+\delta} - (\tau_2(\delta)-1)^{\frac{1}{2}+\delta}\right] + |\Phi||\mathcal{Q}|\frac{2}{\tau_2(\delta)-1} & \text{otherwise} \end{cases},$$

if Algorithm 1 solves $\min_T \mathbb{E}_{\phi\sim\pi^*_t}\left[N(T,\phi)\right]$ *using the greedy algorithm developed in [2].*

In summary, $\forall 0 < \delta < \frac{1}{2}$, Theorem 2 states that if Algorithm 1 solves the CE optimization problem $\min_T \mathbb{E}_{\phi\sim\pi^*_t}\left[N(T,\phi)\right]$ exactly, then $\forall\tau \ge \tau_1(\delta)$, the classical cumulative regret satisfies $\mathbf{Reg}(\tau, 1) = O(\tau^{\frac{1}{2}+\delta})$; on the other hand, if Algorithm 1 solves the CE optimization problem approximately based on the greedy algorithm proposed in [2], then the cumulative scaled regret $\mathbf{Reg}\left(\tau, 1-\ln\left(\pi^*_{\min}\right)\right) = O(\tau^{\frac{1}{2}+\delta})$, $\forall\tau \ge \tau_2(\delta)$. In the second case, the choice of scale $1-\ln\left(\pi^*_{\min}\right)$ is based on the performance bound on the greedy algorithm in the oracle case [3].

The proof of Theorem 2 is nontrivial. Due to the limitation of space, we only briefly outline the proof for the first case when the CE optimization problem is exactly solved. The proof for the second case is similar and is omitted here. Roughly speaking, The proof for the first case proceeds as follows: first, we show that if the CE user preference π^*_t is "close" to the true user preference π^* in interaction t, then the performance of $T^*_t \in \arg\min_T \mathbb{E}_{\phi\sim\pi^*_t}\left[N(T,\phi)\right]$ is "near-optimal". Second, we show that based on Hoeffding's inequality, with "high probability", π^*_t is "close" to π^* in interaction t. Combining these two results, we can show that in interaction t, the performance of T^*_t is "near-optimal" with "high probability" and hence the expected one-interaction regret is "small". By carefully bounding all the one-interaction regrets, we can prove the cumulative regret is sublinear.

5 Future work

There are several possible directions to further extend the current work. The first direction is to alleviate the computational burden of deriving the CE user preference in each interaction. We conjecture this can be achieved by assuming a factored user preference π^* with a factored Beta prior belief. Factored user preference is also expected to further improve the current cumulative regret bound. The second direction is to consider the scenario when the system is allowed to ask multi-way (instead of binary) questions. The third direction is to relax the assumption that the user's target item features are drawn i.i.d. from π^*. For example, we might consider the case in which the user's target items are drawn from time-varying distributions, or the case in which the user's target item follows an exogenous Markov chain. It should be pointed out that as long as we assume that the user behavior is not influenced by what the system has done, the solution methods are essentially the same. The only difference is that learning more complicated models takes longer time.

References

[1] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

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

[3] Daniel Golovin and Andreas Krause. Adaptive Submodularity: Theory and Applications in Active Learning and Stochastic Optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

[4] Bruno Jedynak, Peter Frazier, and Raphael Sznitman. Twenty questions with noise: Bayes optimal policies for entropy loss. *Journal of Applied Probability*, 49(1):114–136, 2012.

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