Adaptive Cascade Submodular Maximization

Shaojie Tang
Naveen Jindal School of Management
University of Texas at Dallas
Richardson, TX, USA
shaojie.tang@utdallas.edu

Jing Yuan
Department of Computer Science
University of Texas at Dallas
Richardson, TX, USA
csyuanjing@gmail.com

ABSTRACT

In this paper, we propose and study the cascade submodular maximization problem under the adaptive setting. The input of our problem is a set of items, each item is in a particular state (i.e., the marginal contribution of an item) which is drawn from a known probability distribution. However, we can not know its actual state before selecting it. As compared with existing studies on stochastic submodular maximization, one unique setting of our problem is that each item is associated with a continuation probability which represents the probability that one is allowed to continue to select the next item after selecting the current one. Intuitively, this term captures the externality of selecting one item to all its subsequent items in terms of the opportunity of being selected. Therefore, the actual set of items that can be selected by a policy depends on the specific ordering it adopts to select items, this makes our problem fundamentally different from classical submodular set optimization problems. Our objective is to identify the best sequence of selecting items so as to maximize the expected utility of the selected items. We propose a class of stochastic utility functions, *adaptive cascade submodular functions*, and show that the objective functions in many practical application domains satisfy adaptive cascade submodularity. Then we develop a 0.12 approximation algorithm to the adaptive cascade submodular maximization problem.

KEYWORDS

Adaptive Submodular Maximization; Submodular Sequencing; Cascade Browse Model

ACM Reference Format:
Shaojie Tang and Jing Yuan. 2021. Adaptive Cascade Submodular Maximization. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 9 pages.

1 INTRODUCTION

Submodular maximization has been extensively studied in the literature [10]. Their objective is to select a group of items that maximize a submodular utility function subject to various constraints. Recently, Golovin and Krause [5] propose the problem of adaptive submodular maximization, a natural stochastic variant of the classical submodular maximization. In particular, they assume that each item is associated with a particular state which is drawn from a known distribution, the only way to know an item's state is to select that item. As compared with the classical submodular maximization, feasible solutions are now policies instead of subsets: the action taken in each step depends on the observations from the previous steps. For example, a typical adaptive policy works as follows: in each step, we select an item, get to see its actual state, then adaptively select the next item based on these observations and so on. They show that an adaptive greedy policy achieves a $1 - 1/e$ approximation ratio when maximizing an adaptive submodular and adaptive monotone utility function subject to a cardinality constraint. Our problem setting is similar to theirs in that we are also interested in selecting a sequence of items adaptively so as to maximize the expected utility. However, one unique setting of our model is that each item is assigned a *continuation probability* [3, 14]. The continuation probability of an item, say i, is defined as the probability that one is allowed to select the next item after i is being selected. The probabilistic continuation setting allows us to capture the scenario where the selecting process could be terminated prematurely. It can be seen that selecting an item with low continuation probability decreases the chance of its subsequent items being selected. Therefore, the actual set of items that can be selected by a policy depends on the specific ordering it adopts to select items. Our setting is motivated by many real-world applications in machine learning, economics, and operations management. We next give three real-world examples that fit our setting.

Example 1: Taking sponsored search advertising as one example, one challenge faced by ad-networks is to select a sequence of advertisements to display to an online user. It is often assumed that the visibility of an advertisement is negatively impacted by the appearances of its preceding advertisements, e.g., the user is less likely to view a new ad after viewing too many other advertisements. One common way to capture this effect is to introduce the continuation probability for each advertisement. In particular, Tang et al. [15] assume that the users scan through the ads in order, after viewing one advertisement, users decide probabilistically whether to click it, as well as whether to continue the scanning process with the ad specific continuation probability. As a result, the user could terminate the ad session prematurely in a probabilistic manner. Assume the revenue generated from one click is known, the goal of ad-network is to adaptively select a sequence of ads to maximize the expected revenue.

Example 2: Our second practical application is sequential product recommendation [4]. One important task of online retailers such as Amazon is to recommend a list of products to each online user. Given a list of recommended products, users scan through them in order, after browsing one product, users decide probabilistically whether to browse the next product with a product-specific continuation probability. After browsing some products, users make a purchase decision within these products (including no purchase option). Similar to the first application, the recommendation process stops immediately whenever the user decides not to browse the

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

next product. Assume each product is associated with a revenue, the goal of the online retailer is to adaptively recommend a group of products to maximize the expected revenue.

Example 3: The third example is pool-based active learning [5]. For example, given a set of possible hypotheses about the profile of an online user, we let the user conduct an online survey to rule out inconsistent hypotheses. One popular way to build the online survey is called paging design [14] where only one question is displayed to the user per page. After finishing a question, the user can click the "Next" button to view the next question or click "Exit" to terminate the survey process probabilistically. Our goal is to adaptively choose a sequence of survey questions to infer the user's profile as accurate as possible. In Section 5, we use pool-based active learning as an example application to evaluate the performance of our algorithms.

One crucial point in the above applications is that the value of a group of items depends not only on the items belong to that group, but also on the specific ordering of those items. This makes our problem different from set optimization problems as we seek a best sequence of items while considering the externality of one item to its subsequent items in terms of the chance of being selected. Although sequence selection has attracted increasing attention these days, most of existing results do not apply to our problem. As will be discussed in Section 2, our utility function does not satisfy the property of "non-decreasing", a common assumption made in many existing studies. Moreover, while the majority of prior research considers non-adaptive setting, we focus on the adaptive setting, where we are allowed to dynamically adjust the selecting strategy based on the current observations. To make our problem approachable, we restrict our attention to a class of stochastic utility functions, *adaptive cascade submodular functions*. Intuitively, any adaptive cascade submodular function must satisfy *diminishing return* condition under the adaptive setting. We show that the objective functions in several practical application domains satisfy adaptive cascade submodularity. We propose a simple algorithm that achieves a 0.12 approximation ratio. We conduct extensive experiments to evaluate the performance of our solutions in the context of pool-based active learning. The experiment results validate the theoretical analysis of our algorithm.

2 RELATED WORK

Submodular maximization has been extensively studied in the literature [10]. However, most of existing studies focus on set optimization problems whose objective is to select set of items that maximizes a submodular utility function. Our focus is on identifying the best sequence of items so as to maximize the expected utility. Although our paper focuses on the adaptive setting, we first review some important studies in the filed of non-adaptive sequence selection. Recently, Streeter and Golovin [11] considered the sequence optimization problem prompted by applications such as online resource allocation. They defined the properties of monotonicity and submodularity over sequences instead of sets. Alaei et al. [1] introduced the term of sequence submodularity and sequence-non-decreasing. Tschiatschek et al. [16] and Mitrovic et al. [8] defined the utility of a sequence over the edges of a directed graph connecting the items together with a submodular function. However, their results do not apply to our setting since our utility function does not satisfy the property of "sequence monotonicity". Intuitively, under our setting, adding an item to an existing sequence could decrease the utility of the original sequence. For example, assume there is a sequence S with positive utility, as well as an item i with zero utility and zero continuation probability. Consider a new sequence that concatenates i and S (i is placed ahead of S), it is easy to verify that the utility of the new sequence is zero which is smaller than the utility of S. Zhang et al. [17] propose the concept of string submodularity. They provide a set of data-dependent approximation bounds for a greedy strategy. It turns out that our utility function under the non-adaptive setting is string submodular, however, the worst-case performance of the greedy strategy [17] is arbitrarily bad in our setting. Tang and Yuan [14] propose and study the cascade submodular maximization problem under the non-adaptive setting. They propose a series of approximation algorithms in the context of quiz design. Note that all studies previously mentioned restrict themselves to the non-adaptive setting where a sequence must be selected all at once. Only recently, Mitrovic et al. [9] extend the previous studies to the adaptive setting and propose the concept of adaptive sequence submodularity. They follow [16] to build their basic model. Our study is different from theirs in that our utility function is defined over subsequences, instead of graphs [9], making their results not applicable to our setting.

Our work is also closely related to stochastic submodular maximization [2, 5]. Golovin and Krause [5] extend submodularity to adaptive policies and propose the concept of adaptive submodularity. They show that the greedy adaptive strategy achieves a $1 - 1/e$ approximation ratio for adaptive submodular maximization subject to a cardinality constraint. Tang [12] develops the first approximation algorithm for maximizing a non-monotone adaptive submodular function. In this work, we generalize the concept of adaptive submodularity to functions over sequences instead of sets, and introduce the concept of adaptive cascade submodular functions. As mentioned earlier, our model allows us to capture the scenario where the selecting process could be terminated prematurely. We develop an adaptive policy that achieves a 0.12 approximation ratio for solving sequence selection problems with an adaptive cascade submodular and monotone function.

3 PRELIMINARIES

We first introduce some notations and define the general class of *adaptive cascade submodular* functions. In the rest of this paper, let $[n]$ denote the set $\{1, 2, \cdots, n\}$, and we use $|S|$ to denote the cardinality of a set or a sequence S.

3.1 Items and States

Let E denote the entire set of m items, and each item $i \in E$ is in a particular state that belongs to a set O of possible states. Denote by $\phi : E \to O$ a realization of the states of items. Let $\Phi = \{\Phi_i \mid i \in E\}$ be a random realization where $\Phi_i \in O$ denotes a random realization of i. After selecting i, its actual state Φ_i is discovered. Let $\mathcal{U}$ denote the set of all realizations, we assume there is a known prior probability distribution $p(\phi) = \{\Pr[\Phi = \phi] : \phi \in \mathcal{U}\}$ over realizations. In addition, there is a vector $\delta = \{\delta_i \mid i \in E\}$ where δ_i denotes the *continuation probability* of item $i \in E$, i.e., it represents the probability that one is allowed to continue to select the next item after selecting i. We are interested in selecting a group of items adaptively as follows: we start by selecting the first item, say $i \in E$, observe its state Φ_i, then with probability δ_i, we continue to select the next item and observe its state, otherwise we terminate the selecting process, and so on. During the selecting process, we say the current process is *live* if we can continue to select the next item, otherwise we say this process is *dead*. Thus, the probability of a selecting process to be live after i is being selected is δ_i. After each selection, we denote by a *partial realization* ψ the observations made so far: ψ is a function from some subset (i.e., those items which are selected so far) of E to their states. We define the *domain* of ψ as the subset of items involved in ψ. Given a realization ϕ and a partial realization ψ, we say ψ is consistent with ϕ if they are equal everywhere in the domain of ψ. We write $\phi \sim \psi$ in this case. We say that ψ is a *subrealization* of ψ' if $\mathrm{dom}(\psi) \subseteq \mathrm{dom}(\psi')$ and they are equal everywhere in $\mathrm{dom}(\psi)$. In this case we write $\psi \subseteq \psi'$. We use $p(\phi \mid \psi)$ to denote the conditional distribution over realizations given a partial realization ψ: $p(\phi \mid \psi) = \Pr[\Phi = \phi \mid \Phi \sim \psi]$.

3.2 Policies and Problem Formulation

Any adaptive strategy of selecting items can be represented using a function π from a set of partial realizations to E, specifying which item to select next, if the current selecting process is still live, given the current observations. Given any policy π and realization ϕ, we say π *adopts* $S_{\pi,\phi}$ under realization ϕ if $S_{\pi,\phi}$ is the longest possible sequence of items that can be selected by π under realization ϕ. Intuitively, by following π, one can successfully select all items in $S_{\pi,\phi}$ under ϕ if the selecting process is never dead. By abuse of notation, we use the same notation $S_{\pi,\phi}$ to denote the set of items in $S_{\pi,\phi}$. Given a sequence $S_{\pi,\phi}$, let $S^{(k)}_{\pi,\phi}$ denote the prefix of $S_{\pi,\phi}$ of length k and let $S^k_{\pi,\phi}$ denote the k-th item in $S_{\pi,\phi}$ for any $k \in [m]$. It follows that under realization ϕ, π selects all and only items from $S^{(k)}_{\pi,\phi}$ with probability $(1-\delta_k)\prod_{i \in S^{(k-1)}_{\pi,\phi}} \delta_i$.

For notational convenience, define $S^{(0)}_{\pi,\Phi} = \emptyset$ for any π and Φ. We next introduce a utility function f from a subset of items and their states to a non-negative real number: $f : 2^E \times O^E \to \mathbb{R}_{\geq 0}$. The expected utility of a policy π under realization ϕ is

$$\sum_{k \in [|S_{\pi,\phi}|]} (1-\delta_{S^k_{\pi,\phi}}) \prod_{i \in S^{(k-1)}_{\pi,\phi}} \delta_i f(S^{(k)}_{\pi,\phi}, \phi)$$

Based on this notation, we define the expected utility $f_{avg}(\pi)$ of a policy π as

$$f_{avg}(\pi) = \mathbb{E}_{\Phi \sim p(\phi)}[\sum_{k \in [|S_{\pi,\Phi}|]} (1-\delta_{S^k_{\pi,\Phi}}) \prod_{i \in S^{(k-1)}_{\pi,\Phi}} \delta_i f(S^{(k)}_{\pi,\Phi}, \Phi)] \quad (1)$$

Our goal is to find a policy π^{opt} that maximizes the expected utility:

$$\pi^{opt} \in \arg\max_{\pi} f_{avg}(\pi)$$

3.3 Adaptive Cascade Submodularity and Monotonicity

We first review two concepts which are defined over set functions. For notational convenience, let $h(\psi) = \mathbb{E}_{\Phi \sim p(\phi|\psi)}[f(\mathrm{dom}(\psi), \Phi)]$ denote the utility of $\mathrm{dom}(\psi)$ under partial realization ψ.

Definition 1. *[5][Adaptive Submodularity] A set function f is adaptive submodular with respect to a prior distribution $p(\phi)$, if for any two partial realizations ψ and ψ' such that $\psi \subseteq \psi'$, and any item $i \in E \setminus \mathrm{dom}(\psi')$, the following holds:*

$$\mathbb{E}_{\Phi \sim p(\phi|\psi)}[f(\{i\} \cup \mathrm{dom}(\psi), \Phi)] - h(\psi) \geq \mathbb{E}_{\Phi \sim p(\phi|\psi')}[f(\{i\} \cup \mathrm{dom}(\psi'), \Phi)] - h(\psi') \quad (2)$$

Definition 2. *[5][Adaptive Monotonicity] A set function f is adaptive monotone with respect to a prior distribution $p(\phi)$, if for any partial realization ψ, and any item $i \in E \setminus \mathrm{dom}(\psi)$, the following holds:*

$$\mathbb{E}_{\Phi \sim p(\phi|\psi)}[f(\{i\} \cup \mathrm{dom}(\psi), \Phi)] - h(\psi) \geq 0 \quad (3)$$

We next introduce the notation of *adaptive cascade submodularity*. For any subset of items $V \subseteq E$, let $\Omega(V)$ denote the set of policies which are allowed to select items only from V. It clear that $\Omega(V) \subseteq \Omega(V')$ for any $V \subseteq V'$.

Definition 3 (Adaptive Cascade Submodularity). *A function f is adaptive cascade submodular with respect to a prior distribution $p(\phi)$, if for any two partial realizations ψ and ψ' such that $\psi \subseteq \psi'$, and any subset of items $V \subseteq E \setminus \mathrm{dom}(\psi')$, the following holds for any δ:*

$$\max_{\pi \in \Omega(V)} f_{avg}(\pi \cup \mathrm{dom}(\psi) \mid \psi) - h(\psi) \geq \max_{\pi \in \Omega(V)} f_{avg}(\pi \cup \mathrm{dom}(\psi') \mid \psi') - h(\psi')$$

where $f_{avg}(\pi \cup \mathrm{dom}(\psi) \mid \psi) =$

$$\mathbb{E}_{\Phi \sim p(\phi|\psi)}[\sum_{k \in [|S_{\pi,\Phi}|]} (1-\delta_{S^k_{\pi,\Phi}}) \prod_{i \in S^{(k-1)}_{\pi,\Phi}} \delta_i f(S^{(k)}_{\pi,\Phi} \cup \mathrm{dom}(\psi), \Phi)] \quad (4)$$

denote the conditional expected utility of a policy that first selects $\mathrm{dom}(\psi)$, then runs π, conditioned on a partial realization ψ.

We next show that if f is adaptive monotone and adaptive cascade submodular with respect to a prior distribution $p(\phi)$, then f is adaptive submodular with respect to the same distribution. Although the other direction is not necessarily true, that is, adaptive submodularity does not imply adaptive cascade submodularity, we find that many well studied adaptive submodular functions are also adaptive cascade submodular. In fact, it is easy to show that if the variables $\{\Phi_i \mid i \in E\}$ are independent, then f is adaptive cascade submodular. One such example is sensor placement [7] where deployed sensors are assumed to fail probabilistically and independently. In other applications including influence maximization and pool-based active learning [5] where $\{\Phi_i \mid i \in E\}$ are not independent, their utility functions also satisfy adaptive cascade submodularity.

Lemma 1. *If f is adaptive monotone and adaptive cascade submodular with respect to a prior distribution $p(\phi)$, then f is adaptive submodular with respect to $p(\phi)$.*

Proof: Since f is adaptive cascade submodular, we have

$$\max_{\pi\in\Omega(V)} f_{avg}(\pi\cup \mathrm{dom}(\psi)\mid\psi)-h(\psi)$$
$$\geq \max_{\pi\in\Omega(V)} f_{avg}(\pi\cup \mathrm{dom}(\psi')\mid\psi')-h(\psi') \quad (5)$$

for any two partial realizations ψ and ψ' such that $\psi\subseteq\psi'$, and any subset of items $V\subseteq E$, according to Definition 3. Consider a singleton $V=\{i\}$ for any $i\in E\setminus\mathrm{dom}(\psi')$, the strategy set $\Omega(\{i\})$ contains only two strategies for any partial realization ψ: selecting i or $\emptyset$. Due to f is adaptive monotone, we have $\max_{\pi\in\Omega(\{i\})} f_{avg}(\pi\cup \mathrm{dom}(\psi)\mid\psi)=f_{avg}(\pi\cup\{i\}\mid\psi)$ for any $i\in E\setminus\mathrm{dom}(\psi')$ and partial realization ψ. Condition (5) can be simplified as

$$f_{avg}(\pi\cup\{i\}\mid\psi)-h(\psi)\geq f_{avg}(\pi\cup\{i\}\mid\psi')-h(\psi') \quad (6)$$

for any two partial realizations ψ and ψ' such that $\psi\subseteq\psi'$, and any $i\in E\setminus\mathrm{dom}(\psi')$. According to (4), we have $f_{avg}(\pi\cup\{i\}\mid\psi)=\mathbb{E}_{\Phi\sim p(\phi\mid\psi)}[f(\{i\}\cup\mathrm{dom}(\psi),\Phi)]$ for any partial realization ψ. It follows that (6) can be rewritten as

$$\mathbb{E}_{\Phi\sim p(\phi\mid\psi)}[f(\{i\}\cup\mathrm{dom}(\psi),\Phi)]-h(\psi)$$
$$\geq \mathbb{E}_{\Phi\sim p(\phi\mid\psi')}[f(\{i\}\cup\mathrm{dom}(\psi'),\Phi)]-h(\psi') \quad (7)$$

for any two partial realizations ψ and ψ' such that $\psi\subseteq\psi'$, and any $i\in E\setminus\mathrm{dom}(\psi')$. According to Definition 1, (7) implies that f is adaptive submodular with respect to $p(\phi)$. □

4 THE ADAPTIVE GREEDY PLUS POLICY

In this section, we propose an adaptive policy which achieves a constant approximation ratio for maximizing an adaptive cascade submodular function.

4.1 Technical Lemmas

Before presenting our algorithm, we first provide some additional notations and technical lemmas that will be used to design and analyze the proposed algorithm. We start by introducing the concept of *reachability.*

Definition 4 (Reachability). *Given a sequence of items S and any $k\in[|S|]$, we define the reachability of its k-th item S^k as $\prod_{i\in S^{(k-1)}}\delta_i$, e.g., it represents the probability of S^k being selected given that S is adopted. For notational convenience, assume $\delta_{S^0}=1$.*

Based on the notion of reachability, we next introduce the concepts of *ρ-reachable sequence*, *strongly ρ-reachable sequence*, *ρ-reachable policy*, and *strongly ρ-reachable policy.*

Definition 5 (ρ-reachable Sequence). *For any $\rho\in[0,1]$, we say a sequence S is* ρ-reachable *if the reachability of all items in S is at least ρ, or in equivalent, the reachability of the last item of S is at least ρ, e.g., $\prod_{i\in S^{(|S|-1)}}\delta_i\geq\rho$.*

Based on the above definition, it can be seen that if we adopt a ρ-reachable sequence S, then the entire S can be selected with probability at least ρ.

Definition 6 (ρ-reachable Policy). *For any $\rho\in[0,1]$, we say a policy π is* ρ-reachable *if for any realization ϕ, it holds that $\prod_{i\in S_{\pi,\phi}^{(|S_{\pi,\phi}|-1)}}\delta_i\geq\rho$. That is, π only adopts ρ-reachable sequence. Let $\Omega(\rho)$ denote the set of all ρ-reachable policies.*

Definition 7 (Strongly ρ-reachable Sequence). *For any $\rho\in[0,1]$, we say a sequence S is* strongly ρ-reachable *if $\prod_{i\in S}\delta_i\geq\rho$.*

Definition 8 (Strongly ρ-reachable Policy). *For any $\rho\in[0,1]$, we say a policy π is* strongly ρ-reachable *if for any realization ϕ, it holds that $\prod_{i\in S_{\pi,\phi}}\delta_i\geq\rho$. That is, a strongly ρ-reachable policy only adopts strongly ρ-reachable sequence. Let $\Omega(\rho^+)$ denote the set of all strong ρ-reachable policies.*

We next introduce the term of maximal ρ-reachable sequence.

Definition 9 (Maximal ρ-reachable Sequence). *Fix any $\rho\in[0,1]$. When $\prod_{i\in E}\delta_i<\rho$, we say a sequence S is a maximal ρ-reachable sequence if S is ρ-reachable but not strongly ρ-reachable. When $\prod_{i\in E}\delta_i\geq\rho$, all sequences are considered as maximal ρ-reachable sequences. Let $\mathcal{G}(\rho)$ denote the set of all maximal ρ-reachable sequences.*

Intuitively, when $\prod_{i\in E}\delta_i<\rho$, a maximal ρ-reachable sequence $G\in\mathcal{G}(\rho)$ must satisfy two conditions: 1. the reachability of all items in G is at least ρ, and 2. any item placed after G has reachability less than ρ.

Based on the above notations, we first show that there exists a ρ-reachable policy whose performance is close to the optimal policy.

Lemma 2. *Fix any $\rho\in[0,1]$. If f is adaptive cascade submodular, there is a ρ-reachable policy $\pi\in\Omega(\rho)$ such that $f_{avg}(\pi)\geq(1-\rho)f_{avg}(\pi^{opt})$.*

Proof: The basic idea of our proof is to show that given an optimal policy π^{opt}, we can discard those items whose reachability is small at the cost of a bounded loss. A similar result was proved in [6] for maximizing a linear utility function under the non-adaptive setting.

For any maximal ρ-reachable sequence $G\in\mathcal{G}(\rho)$, let $\Pr[(G,\psi)]$ denote the probability that G is a prefix of some sequence adopted by π^{opt} while the states of G is ψ, thus $\mathrm{dom}(\psi)=G$. Based on this notation, we can represent the expected utility of π^{opt} as follows:

$$f_{avg}(\pi^{opt}) \quad =\textstyle\sum_{G\in\mathcal{G}(\rho)}\sum_{\Psi:\mathrm{dom}(\Psi)=G}\Pr[(G,\Psi)]g(G,\Psi) \quad (8)$$

where

$$g(G,\Psi)=\sum_{k\in[|G|]}(1-\delta_{G^k})(\prod_{i\in G^{(k-1)}}\delta_i)f(G^{(k)},\Psi)$$
$$+(\prod_{i\in G}\delta_i)(\max_{\pi\in\Omega(E\setminus G)} f_{avg}(\pi\cup\mathrm{dom}(\Psi)\mid\Psi)-h(\Psi))$$

denotes the expected utility of π^{opt} conditioned on G is a prefix of some sequence adopted by π^{opt} while the states of G is Ψ. According to Definition 3, we have $\max_{\pi\in\Omega(E\setminus G)} f_{avg}(\pi\cup\mathrm{dom}(\Psi)\mid\Psi)-h(\Psi)\leq\max_{\pi\in\Omega(E\setminus G)} f_{avg}(\pi\mid\emptyset)-f(\emptyset,\emptyset)$ due to $\emptyset\subseteq\Psi$. Moreover, $\max_{\pi\in\Omega(E\setminus G)} f_{avg}(\pi\mid\emptyset)-f(\emptyset,\emptyset)=\max_{\pi\in\Omega(E\setminus G)} f_{avg}(\pi)\leq\max_{\pi\in\Omega(E)} f_{avg}(\pi)=f_{avg}(\pi^{opt})$ where the inequality is due to $\Omega(E\setminus G)\subseteq\Omega(E)$. Then we have

$$\max_{\pi\in\Omega(E\setminus G)} f_{avg}(\pi\cup\mathrm{dom}(\Psi)\mid\Psi)-h(\Psi)\leq f_{avg}(\pi^{opt}) \quad (9)$$

It follows that

$$g(G,\Psi) \leq \sum_{k\in[|G|]}(1-\delta_{G^k})(\prod_{i\in G^{(k-1)}}\delta_i)f(G^{(k)},\Psi) + (\prod_{i\in G}\delta_i)f_{avg}(\pi^{opt})$$
$$\leq \sum_{k\in[|G|]}(1-\delta_{G^k})(\prod_{i\in G^{(k-1)}}\delta_i)f(G^{(k)},\Psi) + \rho f_{avg}(\pi^{opt})$$

where the first inequality is due to (9) and the second inequality is due to G is a maximal ρ-reachable sequence. Then we have

$$f_{avg}(\pi^{opt}) = \sum_{G\in\mathcal{G}(\rho)}\sum_{\Psi:\mathrm{dom}(\Psi)=G}\Pr[(G,\Psi)]g(G,\Psi)$$
$$\leq \sum_{G\in\mathcal{G}(\rho)}\sum_{\Psi:\mathrm{dom}(\Psi)=G}\Pr[(G,\Psi)]$$
$$\times(\sum_{k\in[|G|]}(1-\delta_{G^k})(\prod_{i\in G^{(k-1)}}\delta_i)f(G^{(k)},\Psi)) + \rho f_{avg}(\pi^{opt}) \quad (10)$$

Based on the above results, we next construct a ρ-reachable policy whose expected utility is at least $(1-\rho)f_{avg}(\pi^{opt})$. Assume the optimal policy π^{opt} is given, we run π^{opt} until the last item whose reachability is smaller than ρ is selected or the current selecting process is dead, whichever comes first. It is clear that the above policy is ρ-reachable, and its expected utility is

$$\sum_{G\in\mathcal{G}(\rho)}\sum_{\Psi:\mathrm{dom}(\Psi)=G}\Pr[(G,\Psi)](\sum_{k\in[|G|]}(1-\delta_{G^k})(\prod_{i\in G^{(k-1)}}\delta_i)f(G^{(k)},\Psi))$$

whose value is lower bounded by $(1-\rho)f_{avg}(\pi^{opt})$ according to (10). This finishes the proof of this lemma. □

According to Lemma 2, there exists a ρ-reachable policy whose expected utility is at least $(1-\rho)f_{avg}(\pi^{opt})$, then the optimal ρ-reachable policy has the same performance bound. In particular, let π^{opt1} denote the optimal solution to **P1**, we have the following lemma.

Lemma 3. *Fix any $\rho\in[0,1]$. If f is adaptive cascade submodular, we have $f_{avg}(\pi^{opt1}) \geq (1-\rho)f_{avg}(\pi^{opt})$.*

Lemma 3 allows us to put our focus on solving **P1**.

P1: *Maximize* $f_{avg}(\pi)$
subject to: $\pi\in\Omega(\rho)$

We next introduce a new optimization problem **P2** subject to only adopting strongly ρ-reachable sequences. Let π^{opt2} denote the optimal solution to **P2**.

P2: *Maximize* $f_{avg}(\pi)$
subject to: $\pi\in\Omega(\rho^+)$

Notice that every strongly ρ-reachable sequence is also a ρ-reachable sequence, similarly, every strongly ρ-reachable policy is also a ρ-reachable policy. Therefore, $f_{avg}(\pi^{opt2})$ is upper bounded by $f_{avg}(\pi^{opt1})$. However, we next show that the gap between $f_{avg}(\pi^{opt1})$ and $f_{avg}(\pi^{opt2})$ can be bounded. For notational simplicity, we use $f(i,\Phi)$ to denote $f(\{i\},\Phi)$.

Lemma 4. *Denote by $i^*=\max_{i\in E}\mathbb{E}_{\Phi\sim p(\phi)}[f(i,\Phi)]$ a single item with the maximum expected utility. If f is adaptive cascade submodular, then $f_{avg}(\pi^{opt2})+\mathbb{E}_{\Phi\sim p(\phi)}[f(i^*,\Phi)] \geq f_{avg}(\pi^{opt1})$.*

Proof: Assume the optimal ρ-reachable policy π^{opt1} is given, we next construct a strongly ρ-reachable policy π' as follows: we run π^{opt1} until it violates the strongly ρ-reachable constraint (the item whose addition to the current solution violates the constraint is not selected) or the current selecting process is dead, whichever comes first. Observe that any sequence S of length k can be represented as $S^{(k-1)}\oplus S^k$, where $\oplus$ is the concatenation operator. Assume S is a ρ-reachable sequence, we have $\prod_{i\in S^{(k-1)}}\delta_i \geq \rho$, thus $S^{(k-1)}$ is a strongly ρ-reachable sequence according to Definition 7. Based on this observation, it is easy to verify that given any full realization, π^{opt1} selects at most one more item (the item that violates the strongly ρ-reachable constraint) than π'. Due to the adaptive submodularity of f (Lemma 1), the expected marginal utility of that item is upper bounded by $\mathbb{E}_{\Phi\sim p(\phi)}[f(i^*,\Phi)]$. It follows that $f_{avg}(\pi')+\mathbb{E}_{\Phi\sim p(\phi)}[f(i^*,\Phi)] \geq f_{avg}(\pi^{opt1})$. Because π^{opt2} is the optimal strongly ρ-reachable policy, we have $f_{avg}(\pi^{opt2})+\mathbb{E}_{\Phi\sim p(\phi)}[f(i^*,\Phi)] \geq f_{avg}(\pi')+\mathbb{E}_{\Phi\sim p(\phi)}[f(i^*,\Phi)] \geq f_{avg}(\pi^{opt1})$. This finishes the proof of this lemma. □

According to Definition 8, a policy π is strongly ρ-reachable if $\forall\phi:\prod_{i\in S_{\pi,\phi}}\delta_i\geq\rho$, which is equivalent to $\forall\phi:\sum_{i\in S_{\pi,\phi}}-\log\delta_i\leq-\log\rho$. Replacing $\pi\in\Omega(\rho^+)$ by $\forall\phi:\sum_{i\in S_{\pi,\phi}}-\log\delta_i\leq-\log\rho$, we obtain an alternative formulation **P2.1** of **P2**.

P2.1: *Maximize* $f_{avg}(\pi)$
subject to: $\forall\phi:\sum_{i\in S_{\pi,\phi}}-\log\delta_i\leq-\log\rho$

To facilitate the analysis of our proposed algorithm, we introduce another optimization problem **P3** by replacing the objective function $f_{avg}(\pi)$ in **P2.1** using $\mathbb{E}_{\Phi\sim p(\phi)}[f(S_{\pi,\Phi},\Phi)]$.

P3: *Maximize* $\mathbb{E}_{\Phi\sim p(\phi)}[f(S_{\pi,\Phi},\Phi)]$
subject to: $\forall\phi:\sum_{i\in S_{\pi,\phi}}-\log\delta_i\leq-\log\rho$

Note that $\mathbb{E}_{\Phi\sim p(\phi)}[f(S_{\pi,\Phi},\Phi)]$ is the expected utility of a policy π assuming that the selecting process is never dead. Therefore, if f is adaptive monotone, then $f_{avg}(\pi)\leq\mathbb{E}_{\Phi\sim p(\phi)}[f(S_{\pi,\Phi},\Phi)]$ for any π. For notation simplicity, define $\overline{f}_{avg}(\pi)=\mathbb{E}_{\Phi\sim p(\phi)}[f(S_{\pi,\Phi},\Phi)]$. Then we have the following lemma.

Lemma 5. *Let π^{opt3} denote the optimal solution to **P3**. If f is adaptive monotone, then $\overline{f}_{avg}(\pi^{opt3})\geq f_{avg}(\pi^{opt2})$.*

Algorithm 1 Candidate Policy π_B
```
1: $S=\emptyset; \psi=\emptyset$
2: while the selecting process is live do
3:     let $i_r=\arg\max_{i\in E}\Delta(i\mid\psi)/c(i)$;
4:     add $i_r$ to $S$; $\psi=\psi\cup\{\phi_{i_r}\}$;
5: end while
6: return $S$
```

4.2 Algorithm Design

Now we are ready to present our adaptive greedy plus policy $\pi^{\text{greedy+}}$. Define $c(i)=-\log\delta_i$ as the *virtual* cost of an item $i\in E$. Intuitively, an item with a higher continuation probability has a smaller virtual cost. $\pi^{\text{greedy+}}$ randomly picks one solution from the

Algorithm 2 Restricted Candidate Policy $\pi_B^{\text{restricted}}$

```
1: $S = \emptyset; B = -\log\rho; \psi = \emptyset$
2: while the selecting process is live do
3:     let $i_r = \arg\max_{i\in E} \Delta(i \mid \psi)/c(i)$;
4:     if $B - c(i_r) \geq 0$ then
5:         add $i_r$ to $S$;
6:         $\psi = \psi \cup \{\phi_{i_r}\}; B \leftarrow B - c(i_r)$;
7:     else
8:         add $i_r$ to $S$;
9:         break;
10:    end if
11: end while
12: return $S$
```

two candidates π_A and π_B such that π_A is selected with probability $1-\frac{1}{\rho(1-1/e)+1}$ and π_B is selected with probability $\frac{1}{\rho(1-1/e)+1}$, where the value of ρ will be decided later. We next describe π_A and π_B in details.

Design of π_A. Selecting a singleton i^* with the maximum expected utility as the first item, then follow an arbitrary sequence of the remaining items to select the next item until the selecting process is dead. Clearly, $f_{avg}(\pi_A) \geq \mathbb{E}_{\Phi\sim p(\phi)}[f(i^*, \Phi)]$ due to f is adaptive monotone.

Design of π_B. Selecting items in a greedy manner as follows: In each round r of a live selecting process, π_B selects an item i_r with the largest "benefit-to-cost" ratio

$$i_r = \arg\max_{i\in E} \Delta(i \mid \psi)/c(i)$$

where $\Delta(i \mid \psi) = \mathbb{E}_{\Phi\sim p(\phi|\psi)}[f(i \cup \text{dom}(\psi), \Phi)] - h(\psi)$ denotes the expected marginal benefit of i conditioned on the current partial realization ψ. This process iterates until the current selecting process is dead. A detailed description of π_B is listed in Algorithm 1.

4.3 Performance Analysis

For the purpose of analysis, we first introduce a restricted version $\pi_B^{\text{restricted}}$ (Algorithm 2) of the second candidate solution π_B by enforcing a budget constraint $-\log\rho$ on the total virtual cost of selected items. That is, in each round of a live selecting process, $\pi_B^{\text{restricted}}$ selects an item that maximizes the ratio of the expected marginal benefit to the virtual cost, and this process continues until the total virtual cost of selected items is larger than $-\log\rho$ or the current selecting process is dead. Note that $\pi_B^{\text{restricted}}$ is very similar to the adaptive greedy algorithm proposed in [5] except that $\pi_B^{\text{restricted}}$ is allowed to violate the budget constraint by adding one more item. That is, the first item that violates the budget constraint is also selected by $\pi_B^{\text{restricted}}$. As π_B always selects items no less than $\pi_B^{\text{restricted}}$, π_B can not perform worse than $\pi_B^{\text{restricted}}$ due to the utility function f is adaptive monotone. To analyze the performance of π_B, it suffice to give a lower bound on $f_{avg}(\pi_B^{\text{restricted}})$.

Now we are ready to present the main theorem of this paper.

Theorem 1. *Fix any $\rho \in [0,1]$. If f is adaptive cascade submodular and adaptive monotone, then*

$$f_{avg}(\pi^{\text{greedy+}}) \geq \frac{\rho(1-1/e)(1-\rho)}{\rho(1-1/e)+1} f_{avg}(\pi^{opt})$$

Proof: We first build a relation between $f_{avg}(\pi_B^{\text{restricted}})$ and $\overline{f}_{avg}(\pi^{opt3})$. Because we assume f is adaptive monotone and adaptive cascade submodular, we have f is adaptive submodular according to Lemma 1. Therefore, **P3** is an adaptive submodular maximization problem subject to a budget constraint, where the cost of each item $i \in E$ is $c(i)$ and the budget constraint is $-\log\rho$. According to [5, 13], the "benefit-to-cost" ratio based greedy algorithm achieves approximation ratio $1-1/e^{\frac{l}{B}}$ when maximizing an adaptive submodular and adaptive monotone function, where l is the (expected) actual amount of budget consumed by the algorithm and B is the budget constraint. This ratio is lower bounded by $1-1/e$ when $l \geq B$. In our case, because $\pi_B^{\text{restricted}}$ is allowed to violate the budget constraint by adding one more item to the solution, we have

$$\overline{f}_{avg}(\pi_B^{\text{restricted}}) \geq (1-1/e)\overline{f}_{avg}(\pi^{opt3}) \tag{11}$$

Moreover, because $\pi_B^{\text{restricted}}$ does not violate the budget constraint until the last round, the reachability of every item selected by $\pi_B^{\text{restricted}}$ is lower bounded by ρ. Thus the expected utility of $\pi_B^{\text{restricted}}$ is at least $f_{avg}(\pi_B^{\text{restricted}}) \geq \rho\overline{f}_{avg}(\pi_B^{\text{restricted}})$. It follows that

$$f_{avg}(\pi_B^{\text{restricted}}) \geq \rho(1-1/e)\overline{f}_{avg}(\pi^{opt3}) \tag{12}$$

due to (11). Now we are ready to bound the approximation ratio of $\pi^{\text{greedy+}}$. Let $\alpha(\rho) = \frac{1}{\rho(1-1/e)+1}$ and $\overline{\alpha}(\rho) = 1-\alpha(\rho)$, we have

$$f_{avg}(\pi^{\text{greedy+}}) = \overline{\alpha}(\rho)f_{avg}(\pi_A) + \alpha(\rho)f_{avg}(\pi_B) \tag{13}$$

$$\geq \alpha(\rho)f_{avg}(\pi_B^{\text{restricted}}) + \overline{\alpha}(\rho)\mathbb{E}_{\Phi\sim p(\phi)}[f(i^*, \Phi)] \tag{14}$$

$$\geq \rho(1-\frac{1}{e})\alpha(\rho)\overline{f}_{avg}(\pi^{opt3}) + \overline{\alpha}(\rho)\mathbb{E}_{\Phi\sim p(\phi)}[f(i^*, \Phi)] \tag{15}$$

$$\geq \rho(1-\frac{1}{e})\alpha(\rho)(f_{avg}(\pi^{opt2}) + \mathbb{E}_{\Phi\sim p(\phi)}[f(i^*, \Phi)]) \tag{16}$$

$$\geq \rho(1-\frac{1}{e})\alpha(\rho)f_{avg}(\pi^{opt1}) \tag{17}$$

$$\geq (1-\rho)\rho(1-\frac{1}{e})\alpha(\rho)f_{avg}(\pi^{opt}) \tag{18}$$

(13) is due to π^{greedy^+} randomly picks one between π_A and π_B such that π_A is selected with probability $1-\alpha(\rho)$ and π_B is selected with probability $\alpha(\rho)$; (14) is due to $f_{avg}(\pi_B^{\text{restricted}}) \leq f_{avg}(\pi_B)$ and $f_{avg}(\pi_A) \geq \mathbb{E}_{\Phi\sim p(\phi)}[f(i^*, \Phi)]$; (15) is due to (12); (16) is due to Lemma 5; (17) is due to Lemma 4; (18) is due to Lemma 3. This finishes the proof of this theorem. □

It can be seen that if we set $\rho = \frac{\sqrt{e(2e-1)}-e}{e-1}$, then $(1-\rho)\rho(1-1/e)\alpha(\rho) > 0.12$. Hence, $\pi^{\text{greedy+}}$ achieves a 0.12 approximation ratio at $\rho = \frac{\sqrt{e(2e-1)}-e}{e-1}$.

Corollary 2. *If we set $\rho = \frac{\sqrt{e(2e-1)}-e}{e-1}$, then our adaptive greedy plus policy $\pi^{\text{greedy+}}$ achieves a 0.12 approximation ratio given that f is adaptive cascade submodular and adaptive monotone.*

5 PERFORMANCE EVALUATION

In this section, we conduct experiments to evaluate the performance of our proposed algorithm in the context of active learning. Assume that we are given a set of hypotheses H, and a set of unlabeled data points X where each $x \in X$ is drawn independently

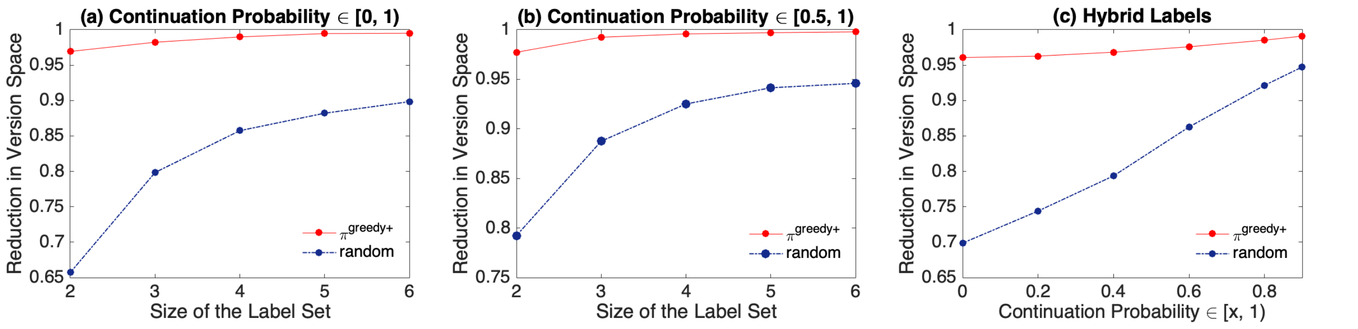

Figure 1: Reduction in version space vs. number of possible labels for each query.

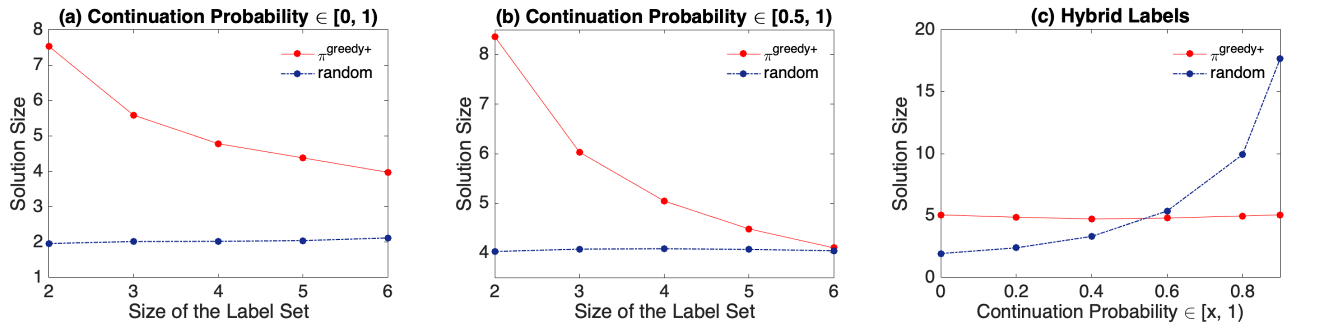

Figure 2: (a) Reduction in version space vs. lower end of the sample range of the continuation probability for each query; and (b) Statistics of the percentage of the selected queries in each group.

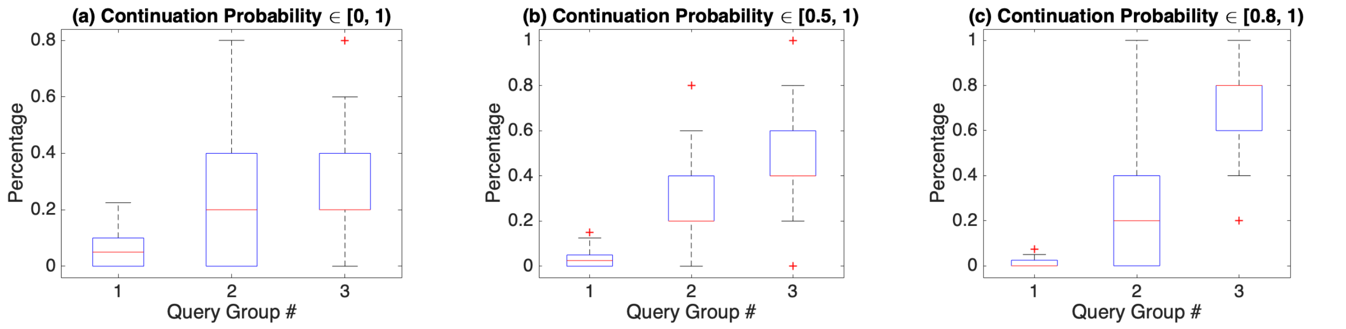

Figure 3: Statistics of the percentage of the selected queries in each group.

from some distribution D. In pool-based active learning, in order to avoid the cost of obtaining labeled data from domain experts, we adaptively identify a sequence of queries that labels a few unlabeled examples until the labels of all unlabeled examples are implied by the obtained labels. Each query is associated with a continuation probability. In the context of online survey (the third example introduced in Section 1), the continuation probability of a query (a survey question) represents the likelihood of a user continuing to answer the next query after answering the current one.

We define the version space to be the set of hypotheses consistent with the observed labels. Intuitively our goal is to reduce the probability mass of the version space as much as possible. Reducing the version space amounts to filtering false hypotheses with stochastic queries. Note query $x \in X$ filters all hypotheses that disagree with the target hypotheses h^* at x. The reduction in version space mass is shown to be adaptive submodular [5]. When running $\pi^{\text{greedy+}}$, we measure the conditional marginal utility of a query as the expected reduction in version space based on the observation of previous requested labels.

In our experiments, we consider 1000 hypotheses with 50 unlabeled data points. For each hypothesis, its probability is drawn from $(0, 1)$ uniformly at random with unity generalization; each data point is assigned a value randomly selected from its possible set of labels. All experiments were run on a machine with Intel Xeon 2.40GHz CPU and 64GB memory, running 64-bit RedHat Linux server. For each set of experiments, we run the simulation for $1,000$ rounds and report the average results as follow.

Our first set of experiments evaluate the performance of our algorithm as measured by the yielded reduction in version space with respect to the changes in the number of possible labels for each data point. We use a random algorithm as our baseline. The random algorithm outputs a sequence of queries in an adaptive manner where each query is randomly selected until the current selecting process is dead. We consider the scenario where each data point has the same number of possible labels. We vary the size of the label set and measure how the performance of our algorithm changes under different settings.

The results are plotted in Figure 1(a) and 1(b). As shown in the figures, the x-axis refers to the size of the label set, ranging from two to six. The y-axis refers to the reduction in version space generated by the corresponding algorithms. We consider two settings of the continuation probability for each query. Figure 1(a) shows the results where each query is assigned a continuation probability drawn uniformly at random from $[0, 1)$. We observe that as the size of the label set increases, the reduction in version space increases for both algorithms. We also observe that $\pi^{\text{greedy+}}$ yields 96.931% reduction in version space for binary labels. It achieves 99.505% reduction when data points have 6 possible labels. It also significantly outperforms the baseline random algorithm in all test cases. The random algorithm only yields 65% reduction in the case of binary labels. Note that our algorithm considers the marginal utility as well as the continuation probability for each query, leading to a higher reduction in version space. In Figure 1(b), each query is assigned a continuation probability drawn uniformly at random from $[0.5, 1)$. Similarly, we observe that $\pi^{\text{greedy+}}$ yields 97.729% reduction in the case of binary labels, and achieves 99.786% reduction when data points have 6 possible labels. Again it outperforms the baseline significantly.

Our second set of experiments explore the impact of the continuation probability of the queries on the reduction in version space, as illustrated in Figure 1(c). We consider the scenario where data points have various numbers of possible labels. We randomly divide our 50 unlabeled data points into three groups. The first group contains 40 data points with binary labels. The second group contains 5 data points with three possible labels. The third group contains 5 data points with four possible labels. We vary the lower end of the interval from which we sample the continuation probability of queries, ranging from 0 to 0.9. Specifically, if the lower end is set to 0, we sample the continuation probability uniformly at random from $[0, 1)$; if the lower end is set to 0.6, we sample the continuation probability uniformly at random from $[0.6, 1)$.

As shown in Figure 1(c), the x-axis holds the lower end of the sample interval, and the y-axis holds the reduction in version space generated by the corresponding algorithms. We observe that as expected, the reduction in version space increases as the lower end of the sample interval increases. Intuitively a larger lower end indicates that the continuation probability is sampled from a pool of larger values. Therefore more queries can be selected in the output sequence, leading to a higher reduction in version space.

Our next set of experiments evaluate how the solution size changes with respect to the changes in the size of the label set, under the scenario where each data point has the same number of possible labels. We vary the size of the label set and measure how the solution size changes under different settings. Figure 2(a) and 2(b) show the results where each query is assigned a continuation probability drawn uniformly at random from $[0, 1)$ and $[0.5, 1)$, respectively. We observe that as the size of the label set increases, the size of the solution generated by $\pi^{\text{greedy+}}$ decreases. We also observe that the size of the solution generated by the random algorithm does not change with the size of the label set.

Next we explore the impact of the continuation probability of the queries on the solution size, as illustrated in Figure 2(c). We use the same setting as in Figure 1(c). We observe that as the sample range of the continuation probability narrows down to the large values, the size of the solution generated by the random algorithm increases rapidly. For $\pi^{\text{greedy+}}$, the solution size does not change much with the sample range.

We take a closer look at the composition of the queries in the output sequence generated by $\pi^{\text{greedy+}}$. As aforementioned we have three groups of queries. We measure the percentage of selected queries in each group. For example, out of 40 data points with binary labels, if 2 of them are selected, then $2/40 = 5\%$ of the queries in group 1 are selected. We plot the statistics for each query group in Figure 3. The x-axis refers to the group number of the queries, and the y-axis refers to the percentage of the selected queries in the corresponding group. Figure 3(a), 3(b) and 3(c) plot the results as the continuation probability of each query is sampled from $[0, 1)$, $[0.5, 1)$, $[0.8, 1)$, respectively. We observe that as the sample range narrows down to the larger values, more queries in group 3, and less queries in group 1, are selected. The reason is that $\pi^{\text{greedy+}}$ takes into account both conditional marginal utility and the continuation probability for each candidate query. For the case with sample interval $[0, 1)$, while queries with four possible labels tend to yield a higher marginal utility, all of them may not be associated with a large continuation probability. Choosing one with small continuation probability may diminish the chance of choosing more queries to further reduce the version space. Therefore some queries with four possible labels are displaced in the output sequence in favor of queries with binary labels and large continuation probability. As the lower limit of the continuation probability goes up, more queries with four possible labels are included in the output sequence.

6 CONCLUSION

In this paper, we propose and study a new stochastic optimization problem, called adaptive cascade submodular maximization. Our goal is to adaptively select a sequence of items that maximizes the expected utility. Our problem is motivated by many real-world applications where the selecting process could be terminated prematurely. We show that existing studies on submodular maximization do not apply to our setting. We start by introducing a class of stochastic utility functions, adaptive cascade submodular functions. Then we propose an adaptive policy that achieves a constant approximation ratio given that the utility function is adaptive cascade submodular and adaptive monotone. In the future, we would like to extend this work by incorporating some practical constraints such as cardinality constraint to the existing model.

REFERENCES

[1] Saeed Alaei, Ali Makhdoumi, and Azarakhsh Malekian. 2010. Maximizing sequence-submodular functions and its application to online advertising. *arXiv preprint arXiv:1009.4153* (2010).

[2] Arash Asadpour and Hamid Nazerzadeh. 2016. Maximizing stochastic monotone submodular functions. *Management Science* 62, 8 (2016), 2374–2391.

[3] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*. 87–94.

[4] Alvaro Flores, Gerardo Berbeglia, and Pascal Van Hentenryck. 2019. Assortment optimization under the sequential multinomial logit model. *European Journal of Operational Research* 273, 3 (2019), 1052–1064.

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

[6] David Kempe and Mohammad Mahdian. 2008. A cascade model for externalities in sponsored search. In *International Workshop on Internet and Network Economics*. Springer, 585–596.

[7] Andreas Krause and Carlos Guestrin. 2007. Near-optimal observation selection using submodular functions. In *AAAI*, Vol. 7. 1650–1654.

[8] Marko Mitrovic, Moran Feldman, Andreas Krause, and Amin Karbasi. 2018. Submodularity on hypergraphs: From sets to sequences. *arXiv preprint arXiv:1802.09110* (2018).

[9] Marko Mitrovic, Ehsan Kazemi, Moran Feldman, Andreas Krause, and Amin Karbasi. 2019. Adaptive sequence submodularity. In *Advances in Neural Information Processing Systems*. 5352–5363.

[10] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions-I. *Mathematical programming* 14, 1 (1978), 265–294.

[11] Matthew Streeter and Daniel Golovin. 2009. An online algorithm for maximizing submodular functions. In *Advances in Neural Information Processing Systems*. 1577–1584.

[12] Shaojie Tang. 2020. Beyond pointwise submodularity: Non-monotone adaptive submodular maximization in linear time. *Theoretical Computer Science* 850 (2020), 249–261.

[13] Shaojie Tang and Jing Yuan. 2020. Influence maximization with partial feedback. *Operations Research Letters* 48, 1 (2020), 24–28.

[14] Shaojie Tang and Jing Yuan. Forthcoming. Cascade Submodular Maximization: Question Selection and Sequencing in Online Personality Quiz. *Production and Operations Management* (Forthcoming).

[15] Shaojie Tang, Jing Yuan, and Vijay Mookerjee. 2020. Optimizing ad allocation in mobile advertising. In *Proceedings of the Twenty-First International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*. 181–190.

[16] Sebastian Tschiatschek, Adish Singla, and Andreas Krause. 2017. Selecting sequences of items via submodular maximization. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[17] Zhenliang Zhang, Edwin KP Chong, Ali Pezeshki, and William Moran. 2015. String submodular functions with curvature constraints. *IEEE Trans. Automat. Control* 61, 3 (2015), 601–616.