Probably Approximately Correct Greedy Maximization

Yash Satsangi
University of Amsterdam
y.satsangi@uva.nl

Shimon Whiteson
University of Oxford
shimon.whiteson@cs.ox.ac.uk

Frans A. Oliehoek
University of Amsterdam
University of Liverpool
frans.oliehoek@liverpool.ac.uk

Abstract

Submodular function maximization finds application in a variety of real-world decision-making problems. However, most existing methods, based on greedy maximization, assume it is computationally feasible to evaluate F, the function being maximized. Unfortunately, in many realistic settings F is too expensive to evaluate exactly even once. We present *probably approximately correct greedy maximization*, which requires access only to cheap anytime confidence bounds on F and uses them to prune elements. We show that, with high probability, our method returns an approximately optimal set. We propose novel, cheap confidence bounds for *conditional entropy*, which appears in many common choices of F and for which it is difficult to find unbiased or bounded estimates. Finally, results on a real-world dataset from a multi-camera tracking system in a shopping mall demonstrate that our approach performs comparably to existing methods, but at a fraction of the computational cost.

1 Introduction

Submodularity is a property of set functions that formalizes the notion of *diminishing returns* i.e., adding an element to a set increases the value of the set function by a smaller or equal amount than adding that same element to a subset. Many real-world problems involve maximizing submodular functions, e.g., summarizing text [Li *et al.*, 2012; Lin and Bilmes, 2010], selecting subsets of training data for classification [Chen and Krause, 2013], or selecting sensors to minimize uncertainty about a hidden variable [Satsangi *et al.*, 2015].

Formally, given a ground set $\mathcal{X} = \{1, 2 \dots n\}$, a set function $F : 2^{\mathcal{X}} \to \mathbb{R}$, is submodular if for every $\mathcal{A}_M \subseteq \mathcal{A}_N \subseteq \mathcal{X}$ and $i \in \mathcal{X} \setminus \mathcal{A}_N$,

$$\Delta_F(i|\mathcal{A}_M) \geq \Delta_F(i|\mathcal{A}_N), \tag{1}$$

where $\Delta_F(i|\mathcal{A}) = F(\mathcal{A} \cup i) - F(\mathcal{A})$ is the *marginal gain* of adding i to A. Typically, the aim is to find an $\mathcal{A}^*$ that maximizes F subject to certain constraints. Here, we consider a constraint on $\mathcal{A}^*$'s size: $\mathcal{A}^* = \arg\max_{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k} F(\mathcal{A})$.

As n increases, the $\binom{n}{k}$ possibilities for $\mathcal{A}^*$ grow rapidly, rendering naive maximization intractable. Instead, *greedy maximization* finds an approximate solution $\mathcal{A}^G$ faster by iteratively adding to a partial solution the element that maximizes the marginal gain. Nemhauser *et al.* (1978) showed that the value obtained by greedy maximization is close to that of full maximization, i.e., $F(\mathcal{A}^G) \geq (1 - e^{-1})F(\mathcal{A}^*)$, if F is submodular, non-negative and monotone.

Lazy greedy maximization [Minoux, 1978] accelerates greedy maximization by pruning elements whose marginal gain on the last iteration ensures that their marginal gain on the current iteration cannot be maximal. *Lazier greedy maximization* [Mirzasoleiman *et al.*, 2015] provides further speedup by evaluating the marginal gain only of a randomly sampled subset of elements at each iteration. Other variations [Wei *et al.*, 2014; Badanidiyuru and Vondrák, 2014] also minimize the number of marginal gain computations.

However, these methods assume it is computationally feasible to exactly compute F, and thus the marginal gain. In many settings, this is not the case. For example, consider a surveillance task [Satsangi *et al.*, 2015] in which an agent aims to minimise uncertainty about a hidden state by selecting a subset of sensors that maximise *information gain*. Computing information gain is computationally expensive, especially when the hidden variable can take many values, as it involves an expectation over the entropy of posterior beliefs about the hidden variable. When surveilling large areas like shopping malls, exactly computing the entropy of a single posterior belief becomes infeasible, let alone an expectation over them.

In this paper, we present a new algorithm called *probably approximately correct greedy maximization*. Rather than assuming access to F itself, we assume access only to confidence bounds on F. In particular, we assume that these bounds are cheaper to compute than F and are *anytime*, i.e., we can tighten them by spending more computation time, e.g., by generating additional samples. Inspired by lazy greedy maximization, our method uses confidence bounds to prune elements, thereby avoiding the need to further tighten their bounds. Furthermore, we provide a PAC analysis that shows that, with high probability, our method returns an approximately optimal set.

Given an unbiased estimator of F, it is possible to use concentration inequalities like Hoeffding's inequality to obtain the confidence bounds needed by PAC greedy maximization. Unfortunately, many applications, such as sensor placement and decision tree induction require information-theoretic def-

initions of F such as information gain that depend on computing entropy over posterior beliefs, which are impossible to estimate in unbiased way [Paninski, 2003]. The absence of an unbiased estimator makes it hard to obtain computationally *cheap* confidence bounds on conditional entropy [Nowozin, 2012; Loh and Nowozin, 2013]. Therefore, in this paper, we propose novel, cheap confidence bounds on conditional entropy.

Finally, we apply PAC greedy maximization with these new confidence bounds to a real-life dataset collected by agents controlling a multi-camera tracking system employed in a shopping mall. Our empirical results demonstrate that our approach performs comparably to greedy and lazier greedy maximization, but at a fraction of the computational cost, leading to much better scalability.

2 Background

Given a set function $F : 2^{\mathcal{X}} \to \mathbb{R}$, *greedy maximization* [Nemhauser *et al.*, 1978] computes a subset $\mathcal{A}^G \subseteq \mathcal{X}$ that approximates $\mathcal{A}^* = \arg\max_{\mathcal{A}\in\mathcal{A}^+} F(\mathcal{A})$, where $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$. As shown in Algorithm 1, it does so by repeatedly adding to $\mathcal{A}^G$ the element i that maximizes the marginal gain $\Delta_F(i|\mathcal{A}^G)$. Because it is greedy, this method is much faster than naive maximization.

Algorithm 1 `greedy-max`$(F, \mathcal{X}, k)$

$\mathcal{A}^G \leftarrow \emptyset$
for $m = 1$ *to* k **do**
  $\mathcal{A}^G \leftarrow \mathcal{A}^G \cup \arg\max_{i\in\mathcal{X}\setminus\mathcal{A}^G} \Delta_F(i|\mathcal{A}^G)$
end for
return $\mathcal{A}^G$

Nemhauser *et al.* (1978) showed that, under certain conditions, this method has bounded error.

Theorem 1. *(Nemhauser et al., 1978) If F is non-negative, monotone and submodular, then $F(\mathcal{A}^G) \geq (1 - e^{-1})F(\mathcal{A}^*)$.*

Lazy greedy maximization [Minoux, 1978] accelerates greedy maximization by pruning elements whose marginal gain cannot be maximal by maintaining a priority queue of all elements in which each element's priority is its marginal gain computed in the *previous* iteration. If in the current iteration, the marginal gain of the element with highest priority is higher than the priority of next element, then the current iteration is terminated since submodularity guarantees that the marginal gain of the remaining elements can only decrease. Lazy greedy maximization computes the same $\mathcal{A}^G$ as greedy maximization and is much faster in practice.

3 Problem Setting

In this paper, we consider a variation on submodular function maximization in which evaluating F, and therefore the marginal gain, is prohibitively expensive, rendering greedy and lazy greedy maximization inapplicable. Instead, we assume access to computationally cheap confidence bounds on F. In particular, let U and L be set functions such that for each $\mathcal{A} \in \mathcal{A}^+$, with probability $1 - \delta_u$, $U(\mathcal{A}) \geq F(\mathcal{A})$ and for each $\mathcal{A} \in \mathcal{A}^+$, with probability $1 - \delta_l$, $F(\mathcal{A}) \geq L(\mathcal{A})$. Furthermore, we assume that U and L are *anytime*, i.e., we have a `tighten` procedure that improves these bounds in exchange for computation. Specifically, calling `tighten`$(\mathcal{A})$ will in expectation reduce $U(\mathcal{A})$ and increase $L(\mathcal{A})$.

These assumptions are easily satisfied in many settings where F is too expensive to compute exactly. For example, if $F(\mathcal{A}) = \mathbb{E}[X|\mathcal{A}]$ for some random variable X, then $\hat{F}(\mathcal{A}) = \frac{1}{N}(\sum_{i=1}^{N} x_i)$, where the x_i's are i.i.d. samples of X, is an unbiased estimator of $\hat{F}(\mathcal{A})$ for which U and L can easily be constructed using, e.g., Hoeffding's inequality [Hoeffding, 1963]. Furthermore, `tighten` need only fold more samples into $\hat{F}$. However, we specifically do *not* assume access to $\hat{F}$ or any other unbiased estimator of F. Instead, we seek an algorithm that performs submodular function maximization given only U, L, and `tighten`.

The absence of an unbiased estimator of F arises in many settings in which F is defined using information-theoretic metrics such as *information gain* or *entropy*. For example, consider the *sensor selection* problem [Williams *et al.*, 2007; Spaan and Lima, 2009] in which an agent has a set of sensors $\mathcal{X} = \{1, 2 \dots n\}$ giving information about a hidden state s. For each sensor i, z_i denotes the observation the agent will receive if it selects that sensor, with $z_i = \emptyset$ if not selected. $\mathbf{z} = \langle z_1, z_2 \dots z_n \rangle$ denotes the complete observation vector generated by all sensors.

Upon selecting sensors $\mathcal{A}$ and observing $\mathbf{z}$, the agent can compute a posterior belief using Bayes rule:

$$b^{\mathcal{A}}_{\mathbf{z}}(s) = \frac{1}{\Pr(\mathbf{z}|b, \mathcal{A})}[\Pr(\mathbf{z}|s, \mathcal{A})b(s)], \tag{2}$$

where $\Pr(\mathbf{z}|b, \mathcal{A}) = \sum_s b(s) \Pr(\mathbf{z}|s, \mathcal{A})$ and $b(s)$ is a prior belief. The agent aims to minimize its uncertainty about s, measured as the *entropy* of $b(s)$: $H_b(s) = -\sum_s b(s) \log(b(s))$. Given b and $\mathcal{A}$, the *conditional entropy* is:

$$H^{\mathcal{A}}_b(s|\mathbf{z}) = \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) H_{b^{\mathcal{A}}_{\mathbf{z}}}(s), \tag{3}$$

where Ω is the set of all possible values of $\mathbf{z}$ that can come from sensors present in the set $\mathcal{A}$. The agent's goal is to find $\mathcal{A}^*$ that maximizes *information gain*:

$$IG_b(\mathcal{A}) = H_b(s) - H^{\mathcal{A}}_b(s|\mathbf{z}). \tag{4}$$

Since the first term in (4) is independent of $\mathcal{A}$, we equivalently define $F(\mathcal{A})$ as:

$$F(\mathcal{A}) = -\sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) H_{b^{\mathcal{A}}_{\mathbf{z}}}(s). \tag{5}$$

Unfortunately, when there are many possible states and actions, computing $H_{b^{\mathcal{A}}_{\mathbf{z}}}(s)$ is not only intractable, but also difficult to efficiently estimate [Paninski, 2003; Nowozin, 2012; Schürmann, 2004]. In fact, Paninski (2003) showed that no unbiased estimator for entropy exists.

Therefore, in the next section we propose a new fundamentally different method that requires only U, L, and `tighten`. To solve sensor selection in particular, we also need cheap anytime implementations of U and L for conditional entropy, which we propose in Section 6.

4 Method

In this section, we propose *probably approximately correct greedy maximization*, which enables an agent to perform submodular function maximization without ever computing F exactly. The main idea is to use U and L to prune elements that with high probability do not maximize marginal gain.

Our approach is inspired by lazy greedy maximization. To see how, it is helpful to view lazy greedy maximization as a pruning method: terminating an iteration before the priority queue is empty effectively prunes each element whose upper bound (given by its marginal gain on the previous iteration) is lower than the maximum lower bound (given by the best marginal gain found so far on the current iteration).

PAC greedy maximization generalizes this idea in two ways. First, it accepts arbitrary upper and lower bounds. This makes it possible to replace the bounds used by lazy greedy maximization, which rely on exact computation of marginal gain, with cheaper ones. Second, it uses confidence bounds instead of hard bounds. By tolerating a small probability of error, our approach can prune more aggressively, enabling large speedups while maintaining a PAC bound.

Algorithm 2 `pac-greedy-max`$(U, L, \mathcal{X}, k, \epsilon_1, t)$

$\mathcal{A}^P \leftarrow \emptyset$
for $m = 1$ *to* k **do**
  $\mathcal{A}^P \leftarrow \mathcal{A}^P \cup$ `pac-max`$(U, L, \mathcal{A}^P, \epsilon_1, t)$
end for
return $\mathcal{A}^P$

Algorithm 2 shows the main loop, which simply adds at each iteration the element selected by the `pac-max` subroutine. Algorithm 3 shows this subroutine, which maintains a queue of unpruned elements prioritized by their upper bound. In each iteration of the outer while loop, `pac-max` examines each of these elements and prunes it if its upper bound is not at least ϵ_1 greater than the max lower bound found so far. In addition, the element with the max lower bound is never pruned. If an element is not pruned, then its bounds are tightened. Algorithm 3 terminates when only one element remains or when the improvement produced by tightening U and L falls below a threshold t.

Algorithm 3 is closely related to *best arm identification* algorithms [Audibert and Bubeck, 2010] for multi-armed bandits. The main difference is that it does not directly estimate F. Instead, it selects an element using only U and L. In cases where an unbiased sample-based estimator of F is available, `pac-max` can be replaced by an off-the-shelf best arm identification algorithm that uses, e.g., Hoeffding bounds instead of U and L. However, in this paper, we focus on the setting in which no such estimator is available and thus bandit-based algorithms cannot be directly applied.

5 PAC Bounds

In this section, we analyze PAC greedy maximization. With oracle access to F, greedy maximization is guaranteed to find $\mathcal{A}^G$ such that $F(\mathcal{A}^G) \geq (1 - e^{-1})F(\mathcal{A}^*)$, if F is monotone, non-negative and submodular [Nemhauser *et al.*, 1978].

Algorithm 3 `pac-max`$(U, L, \mathcal{A}^P, \epsilon_1, t)$

$i^P \leftarrow 0$ % element with max lower bound
for $i \in \mathcal{X} \setminus \mathcal{A}^P$ **do**
  ρ.`enqueue`$(i, U(\mathcal{A}^P \cup i))$ % initial priority
  $i^P \leftarrow \arg\max_{j \in \{i, i^P\}} L(\mathcal{A}^P \cup j)\}$
end for
while ρ.`length`$() > 1 \vee$ change in U & L is $< t$ **do**
  $\rho' \leftarrow$ empty queue
  while $\neg\rho$.`empty`$()$ **do**
    $i \leftarrow \rho$.`dequeue`$()$
    if $i = i^P \vee U(\mathcal{A}^P \cup i) >= L(\mathcal{A}^P \cup i^P) + \epsilon_1$ **then**
      `tighten`$(\mathcal{A}^P \cup i)$
      $i^P \leftarrow \arg\max_{j \in \{i, i^P\}} L(\mathcal{A}^P \cup j)$
      ρ'.`enqueue`$(i, U(\mathcal{A}^P \cup i))$
    end if
  end while
  $\rho \leftarrow \rho'$
end while
return i^P

Since PAC greedy maximization does not assume oracle access to F and instead works with cheap anytime confidence bounds on F, we prove a PAC bound for PAC greedy maximization. In particular, we prove that under the same conditions, PAC greedy maximization finds a solution $\mathcal{A}^P$ such that, with high probability $F(\mathcal{A}^P)$ is close to $F(\mathcal{A}^*)$.

Since greedy maximization does not assume oracle access to marginal gain, it is fundamentally different from previous approaches to greedy maximization [Minoux, 1978; Mirzasoleiman *et al.*, 2015], which are typically analysed in terms of number of evaluations of F to find $\mathcal{A}^G$. Such an analysis is simply not applicable to PAC greedy maximization as it makes no queries to F, instead it repeatedly calls U, L and `tighten`. A first reaction might be to analyze the algorithm in terms of these calls, but we argue this would not prove useful. The number of such calls is highly dependent on the nature of F and `tighten`, which means that it is not possible to quantify the number of calls without making very strong assumptions. For example, if we assume that, after at most T `tighten` calls, both U and L converge to F, the number of `tighten` calls is $\mathcal{O}(nT)$, since in the worst case we need to tighten all elements to their true value. However, such an analysis is not helpful since we typically cannot quantify T. On the contrary, PAC greedy maximization would work in a much more general setting than such assumptions would permit. In fact, PAC greedy maximzation can still be employed even when there is no unbiased estimator of F available, which is a common assumption [Krause and Guestrin, 2005; Bubeck and Cesa-Bianchi, 2012] for such settings. As we show in Section 6, for conditional entropy for which unbiased estimator is not available, even defining `tighten`, much less quantifying T, is challenging.

Thus, in this section, we present our analysis of PAC greedy maximization without making any specific assumptions about U, L, or `tighten`, except the following:

Assumption 1. `pac-max` always terminates with $\rho.length() = 1$.

In other words, we assume that U and L can be tightened enough to disambiguate i^P. If U and L converge to F and sufficient computation time is available, this assumption is easily satisfied. In Section 8, we show that PAC greedy maximization performs well even when Assumption 1 is violated.

We can now prove a lemma that shows that with high probability the marginal gain of the element picked by $\texttt{pac-max}(U, L, \mathcal{A}, \epsilon_1)$ is at least nearly as great as the average marginal gain of the elements not in $\mathcal{A}$.

Lemma 1. *For all $\mathcal{A} \in \mathcal{A}^+$, if $i^P = \texttt{pac-max}(U, L, \mathcal{A}, \epsilon_1, t)$ and Assumption 1 holds, then with probability $1 - \delta_1$,*

$$\Delta_F(i^P|\mathcal{A}) \geq \frac{1}{k} \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}} \Delta_F(i|\mathcal{A}) - \epsilon_1, \tag{6}$$

where $\delta_1 = \delta_u + \delta_l$.

Proof. Let $i^* = \arg\max_{i \in \mathcal{X} \setminus \mathcal{A}} \Delta_F(i|\mathcal{A})$. If $i^* = i^P$, then Lemma 1 holds trivially. If $i^P \neq i^*$, and $\rho.length() = 1$, then Algorithm 3 must have pruned i^* such that: $L(\mathcal{A} \cup i^P) > U(\mathcal{A} \cup i^*) - \epsilon_1$. Since $F(\mathcal{A} \cup i^P) \geq L(\mathcal{A} \cup i^P)$ is true with probability $1 - \delta_l$ and with probability $1 - \delta_u, U(\mathcal{A} \cup i^*) \geq F(\mathcal{A} \cup i^*)$, then using a union bound, with probability $1 - \delta_1$,

$$F(\mathcal{A} \cup i^P) > F(\mathcal{A} \cup i^*) - \epsilon_1, \tag{7}$$

which implies, $\Delta_F(i^P|\mathcal{A}) > \Delta_F(i^*|\mathcal{A}) - \epsilon_1$. □

Theorem 2. *If F is non-negative, monotone and submodular in $\mathcal{X}$ and if Assumption 1 holds then, with probability $1 - \delta$,*

$$F(\mathcal{A}^P) \geq (1 - e^{-1}) F(\mathcal{A}^*) - \epsilon, \tag{8}$$

where $\mathcal{A}^P = \texttt{pac-greedy-max}(U, L, \mathcal{X}, k, \epsilon_1, t)$, $\mathcal{A}^* = \arg\max_{\mathcal{A} \in \mathcal{A}^+} F(\mathcal{A})$, $\delta = k\delta_1$, and $\epsilon = k\epsilon_1$.

Proof. Let $\mathcal{A}^* = \{i_1^*, i_2^*, \ldots, i_k^*\}$ and $\mathcal{A}_m^P = \{i_1^P, i_2^P, \ldots, i_m^P\}$ be the solution returned by Algorithm 2 after $m \leq k$ iterations. Let $\mathcal{A}^S = \mathcal{A}^* \setminus \mathcal{A}_m^P = \{i_1^S, \ldots, i_j^S\}$ and let $\mathcal{A}_l^S$ be the first l elements of $\mathcal{A}^*$, with $\mathcal{A}_0^S = \emptyset$. Note that $F(\mathcal{A}^* \cup \mathcal{A}_m^P)$ can be expressed as:

$$F(\mathcal{A}^* \cup \mathcal{A}_m^P) = F(\mathcal{A}_m^P) + \sum_{l=1}^{j} \Delta_F(i_l^S | \mathcal{A}_m^P \cup \mathcal{A}_{l-1}^S\}). \tag{9}$$

F is monotonic, $F(\mathcal{A}^* \cup \mathcal{A}_m^P) \geq F(\mathcal{A}^*)$, and by submodularity, $\sum_{l=1}^{j} \Delta_F(i_l^S|\mathcal{A}_m^P) \geq \sum_{l=1}^{j} \Delta_F(i_l^S|\mathcal{A}_m^P \cup \mathcal{A}_{l-1}^S\})$. Thus,

$$F(\mathcal{A}_m^P) + \sum_{i \in \mathcal{A}^S} \Delta_F(i|\mathcal{A}_m^P) \geq F(\mathcal{A}^*). \tag{10}$$

Applying Lemma 1 implies that with probability $1 - \delta_1$,

$$F(\mathcal{A}_{m+1}^P) - F(\mathcal{A}_m^P) \geq \frac{1}{k}[F(\mathcal{A}^*) - F(\mathcal{A}_m^P)] - \epsilon_1, \tag{11}$$

By mathematical induction on m, (8) can be obtained. □

Theorem 2 proves that PAC greedy maximization, while assuming access only to anytime confidence bounds on F, computes $\mathcal{A}^P$ such that with high probability $F(\mathcal{A}^P)$ has bounded error with respect to $F(\mathcal{A}^*)$. As PAC greedy maximization requires access to cheap upper and lower confidence bounds, in the next section, we propose such bounds for conditional entropy.

6 Conditional Entropy Bounds

In many settings, U and L can easily be constructed using, e.g., Hoeffding's inequality [Hoeffding, 1963] and `tighten` need only fold more samples into an estimate of F. However, Hoeffding's inequality only bounds the error between the estimate and the expected value of the estimator. This in turn bounds the error between the estimate and the true value only if the estimator is unbiased, i.e., the expected value of the estimator equals the true value.

We are interested in settings such as sensor selection, where F is based on conditional entropy, which is computed by approximating the entropy over a posterior belief that cannot be estimated in an unbiased way [Paninski, 2003]. Therefore, in this section, we propose novel, cheap confidence bounds on conditional entropy. We start by defining the maximum likelihood estimate of entropy. Given M samples, $\{s^1, s^2 \ldots s^M\}$ from a discrete distribution $b(s)$, the *maximum likelihood estimator* (MLE) of $b(s)$ is:

$$\hat{b}(s) = \frac{1}{M} \sum_{j=1}^{M} \mathbb{1}(s^j, s), \tag{12}$$

where $\mathbb{1}(s^j, s)$ is an indicator function that is 1 if $s^j = s$ and 0 otherwise. The MLE of entropy is:

$$H_{\hat{b}}(s) = \sum_s \hat{b}(s) \log(\hat{b}(s)). \tag{13}$$

Though $H_{\hat{b}}(s)$ is known to be biased, Paninski (2003) established some useful properties of it.

Theorem 3. *(Paninski 2003)*

$$(a)\ \Pr(|H_{\hat{b}}(s) - \mathbb{E}[H_{\hat{b}}(s) \mid b]| \geq \eta) \leq \delta_\eta, \tag{14}$$

where $\delta_\eta = 2e^{\frac{-M}{2}\eta^2(\log(M))^{-2}}$.

$$(b)\ \mu_M(b) \leq \mathbb{E}[H_{\hat{b}}(s) \mid b] - H_b(s) \leq 0. \tag{15}$$

where $\mu_M(b) = -\log(1 + \frac{\psi_b(s)-1}{M})$ and $\psi_b(s)$ is the support of $b(s)$.

Hence (14) bounds the variance of $H_{\hat{b}}(s)$ and (15) bounds its bias, which is always negative.

6.1 Lower Confidence Bound

Let $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$ be defined as:

$$H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z}) = \sum_{\mathbf{z}_i \in \Omega} \Pr(\mathbf{z}_i|b, \mathcal{A}) H_{\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}}(s), \tag{16}$$

where $H_{\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}}(s)$ is the MLE of the entropy of the posterior distribution $\hat{b}_{\mathbf{z}}^{\mathcal{A}}(s)$.

Lemma 2. *With probability $1 - \delta_l$,*

$$H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z}) \leq H_b^{\mathcal{A}}(s|\mathbf{z}) + \eta, \tag{17}$$

where $\delta_l = |\Omega|\delta_\eta$.

Proof. For a given $\mathbf{z}_i$, (14) and (15) imply that, with probability $1 - \delta_\eta$,

$$H_{\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}}(s) \leq H_{b_{\mathbf{z}_i}^{\mathcal{A}}}(s) + \eta \tag{18}$$

This is true for each $\mathbf{z}_i \in \Omega$. Taking an expectation over $\mathbf{z}_i$ and using a union bound, yields the final result. □

Typically, the bottleneck in computing $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$ is performing the belief update to find $\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}$ for each $\mathbf{z}_i$. In practice, we approximate these using *particle belief updates* [Doucet *et al.*, 2001], which, for a given $\mathbf{z}_i$, generate a sample s^j from $\hat{b}(s)$ and then an observation $\mathbf{z}'$ from $\Pr(\mathbf{z}|s^j, \mathcal{A})$. If $\mathbf{z}_i = \mathbf{z}'$, then s^j is added to the set of samples approximating $\hat{b}_{\mathbf{z}_i^{\mathcal{A}}}$. Consequently, $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$ can be tightened by increasing M, the number of samples used to estimate $\hat{b}(s)$, and/or increasing the number of samples used to estimate each $\hat{b}_{\mathbf{z}_i^{\mathcal{A}}}$. However, tightening $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$ by using larger values of M is not practical as computing it involves new posterior belief updates (with a larger value of M) and hence increases the computational cost of tightening $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$.

6.2 Upper Confidence Bound

Since $H_{\hat{b}}(s)$ is negatively biased, finding an upper confidence bound is more difficult. A key insight is that such a bound can nonetheless be obtained by estimating posterior entropy using an artificially "coarsened" observation function. That is, we group all possible observations into a set Φ of clusters and then pretend that, instead of observing $\mathbf{z}$, the agent only observes what cluster $\mathbf{z}$ is in. Since the observation now contains less information, the conditional entropy will be higher, yielding an upper bound. Furthermore, since the agent only has to reason about $|\Phi|$ clusters instead of $|\Omega|$ observations, it is also cheaper to compute. Any generic clustering approach, e.g., ignoring certain observation features can be used, though in some cases domain expertise may be exploited to select the clustering that yields the tightest bounds.

Let $\mathbf{r} = \langle r_1 \dots r_n \rangle$ represent a crude approximation of $\mathbf{z}$. That is, for every i, r_i is obtained from z_i by $r_i = f(z_i, d)$, where f clusters z_i into d clusters deterministically and r_i denotes the cluster z_i belongs to. Also, if $z_i = \emptyset$, then $r_i = \emptyset$. Note that $H_b(\mathbf{r}|\mathbf{z}) = 0$ and the domain of r_i and r_j share only $\emptyset$ for all i and j.

Lemma 3. $H_b^{\mathcal{A}}(s|\mathbf{z}) \leq H_b^{\mathcal{A}}(s|\mathbf{r})$.

Proof. Using the chain rule for entropy, on $H_b^{\mathcal{A}}(s, \mathbf{z}|\mathbf{r})$

$$H_b^{\mathcal{A}}(s|\mathbf{z}, \mathbf{r}) + H_b^{\mathcal{A}}(\mathbf{z}|\mathbf{r}) = H_b^{\mathcal{A}}(\mathbf{z}|s, \mathbf{r}) + H_b^{\mathcal{A}}(s|\mathbf{r}). \quad (19)$$

Since $\mathbf{r}$ contains no additional information, $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z}, \mathbf{r}) = H_b^{\mathcal{A}}(s|\mathbf{z})$, and $H_b^{\mathcal{A}}(s|\mathbf{z}) + H_b^{\mathcal{A}}(\mathbf{z}|\mathbf{r}) = H_b^{\mathcal{A}}(\mathbf{z}|s, \mathbf{r}) + H_b^{\mathcal{A}}(s|\mathbf{r})$. Since conditioning can never increase entropy, $H_b^{\mathcal{A}}(\mathbf{z}|s, \mathbf{r}) \leq H_b^{\mathcal{A}}(\mathbf{z}|\mathbf{r})$ [Cover and Thomas, 1991], the stated result holds. □

$H_b^{\mathcal{A}}(s|\mathbf{r})$ is cheaper to compute than $H_b^{\mathcal{A}}(s|\mathbf{z})$ because it requires only $|\Phi|$ belief updates instead of $|\Omega|$. Starting with a small $|\Phi|$, $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r})$ can be tightened by increasing the number of clusters and thus $|\Phi|$.

Note that computing $H_b^{\mathcal{A}}(s|\mathbf{r})$ requires $\Pr(\mathbf{r}|s, \mathcal{A})$, which can be computed by marginalizing $\mathbf{z}$ out from $\Pr(\mathbf{z}|s, \mathcal{A})$, a computationally expensive operation. However, this marginalization only needs to be done once and can be reused when performing greedy maximization for various $b(s)$. This occurs naturally in, e.g., sensor selection, where the hidden state that the agent wants to track evolves over time. At every time step, $b(s)$ changes and a new set $\mathcal{A}^P$ must be selected.

However, computing $H_{b_{\mathbf{r}}^{\mathcal{A}}}(s)$ still requires iterating across all values of s. Thus, to lower the computational cost further, we use estimates of entropy, as with the lower bound:

$$H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r}) = \sum_{\mathbf{r}_i \in \Phi} \Pr(\mathbf{r}_i|b, \mathcal{A}) H_{\hat{b}_{\mathbf{r}_i}^{\mathcal{A}}}(s). \quad (20)$$

Computing $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r})$ is cheaper than $H_b^{\mathcal{A}}(s|\mathbf{r})$ but is not gauranteed to be greater than $H_b^{\mathcal{A}}(s|\mathbf{z})$ since the entropy estimates have negative bias. However, we can still obtain an upper confidence bound.

Lemma 4. *With probability* $1 - \delta_u$

$$H_b^{\mathcal{A}}(s|\mathbf{z}) \leq H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r}) + \eta + \mu_M(b), \quad (21)$$

where $\delta_u = |\Phi|\delta_\eta$.

Proof. (15) implies that, for any fixed $\mathbf{r}_i \in \Phi$,

$$H_{b_{\mathbf{r}_i}^{\mathcal{A}}}(s) \leq \mathbb{E}[H_{\hat{b}_{\mathbf{r}_i}^{\mathcal{A}}}(s) \mid b_{\mathbf{r}_i}^{\mathcal{A}}] + \mu_M(b). \quad (22)$$

Taking an expectation on both sides:

$$\mathbb{E}_{\mathbf{r}_i}[H_{b_{\mathbf{r}_i}^{\mathcal{A}}}(s) \mid b, \mathcal{A}] \leq \mathbb{E}_{\mathbf{r}_i}[\mathbb{E}[H_{\hat{b}_{\mathbf{r}_i}^{\mathcal{A}}}(s) \mid b_{\mathbf{r}_i}^{\mathcal{A}}] \mid b, \mathcal{A}] + \mu_M(b),$$

Now, (14) implies that, with probability $1 - \delta_\eta$,

$$\mathbb{E}[H_{\hat{b}_{\mathbf{r}_i}^{\mathcal{A}}}(s) \mid b_{\mathbf{r}_i}^{\mathcal{A}}] \leq H_{\hat{b}_{\mathbf{r}_i}^{\mathcal{A}}}(s) + \eta. \quad (23)$$

Taking expectations on both sides and using a union bound gives, with probability $1 - \delta_u$,

$$H_b^{\mathcal{A}}(s|\mathbf{r}) \leq H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r}) + \eta + \mu_M(b). \quad \square$$

In practice, we use a larger value of M when computing $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r})$ than $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$. Doing so is critical for reducing the negative bias in $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$. Furthermore, doing so does not lead to intractability because choosing a small $|\Phi|$ ensures that few belief updates will be performed.

Thus, when computing $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$, we set M low but perform many belief updates; when computing $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r})$ we set M high but perform few belief updates. This yields cheap upper and lower confidence bound for conditional entropy.

The following theorem ties together all the results presented in this paper. Note that, since F is defined as *negative* conditional entropy, L is defined using our *upper* bound and U using our *lower* bound.

Theorem 4. *Let* $F(\mathcal{A}) = -H_b^{\mathcal{A}}(s|\mathbf{z})$, $L(\mathcal{A}) = H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r}) + \eta + \log(1 + \frac{1}{M}(\psi_b(s) - 1))$, $U(\mathcal{A}) = H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$ *and* $\mathcal{A}^{\mathcal{P}} =$ `pac-greedy-max`$(U, L, \mathcal{X}, k, \epsilon_1, t)$. *If Assumption 1 holds and* $\mathbf{z}$ *is conditionally independent given* s *then, with probability* $1 - \delta$,

$$F(\mathcal{A}^P) \geq (1 - e^{-1})F(\mathcal{A}^*) - \epsilon, \quad (24)$$

where $\delta = 2k|\Omega|\delta_\eta$, $\epsilon = k\epsilon_1$.

Proof. Note that $\delta_l = |\Phi|\delta_\eta$. Since $\mathbf{z}$ cannot be clustered in more than $|\Omega|$ clusters, $|\Phi| \leq |\Omega|$, thus, $\delta_l \leq |\Omega|\delta_\eta$, $(\delta_l + \delta_u) \leq 2|\Omega|\delta_\eta$, and $k(\delta_l + \delta_u) \leq 2k|\Omega|$. [Krause and Guestrin, 2005] showed that F is submodular if $\mathbf{z}$ is conditionally independent given s. Thus, Theorem 2 with $\epsilon = k\epsilon_1$ and $\delta_1 = \delta_u + \delta_l$ (as per the definition of U and L), implies the stated result. □

7 Related Work

Most work on submodular function maximization focuses on algorithms for approximate greedy maximization that minimize the number of evaluations of F [Minoux, 1978; Wei *et al.*, 2014; Badanidiyuru and Vondrák, 2014; Mirzasoleiman *et al.*, 2015]. In particular, [Mirzasoleiman *et al.*, 2015] randomly sample a subset from $\mathcal{X}$ on each iteration and select the element from this subset that maximizes the marginal gain. Badanidiyuru and Vondrák [Badanidiyuru and Vondrák, 2014] selects an element on each iteration whose marginal gain exceeds a certain threshold. Other proposed methods that maximize surrogate submodular functions [Wei *et al.*, 2014; Chen and Krause, 2013] or address streaming [Krause and Gomes, 2010] or distributed settings [Mirzasoleiman *et al.*, 2013], also assume access to exact F. In contrast, our approach assumes that F is too expensive to compute even once and works instead with confidence bounds on F. Krause and Guestrin (2005) propose approximating conditional entropy for submodular function maximization while still assuming that they can compute the exact posterior entropies. In our case, computing exact posterior entropy is prohibitively expensive.

Streeter and Golovin (2009) and Radlinski *et al.* (2008) propose conceptually related methods that also assume F is never computed exactly. However, their *online* setting is fundamentally different in that the system must first select an entire subset $\mathcal{A} \in \mathcal{A}^+$ and only then receives an estimate of $F(\mathcal{A})$, as well as estimates of the marginal gain of the elements in $\mathcal{A}$. Since the system learns over time how to maximize F, it is a variation on the multi-armed bandit setting. By contrast, we assume that feedback about a given element's marginal gain is available (through tightening U and L) *before* committing to that element.

As mentioned earlier, Algorithm 3 is closely related to *best arm identification* algorithms [Audibert and Bubeck, 2010]. However, such methods assume an unbiased estimator of F is available and hence concentration inequalities like Hoeffding's inequality are applicable. An exception is the work of Loh and Nowozin (2013), which bounds the difference between an entropy estimate and that estimate's expected value. However, since the entropy estimator is biased, this does not yield confidence bounds with respect to the true entropy. While they propose using their bounds for best arm identification, no guarantees are provided, and would be hard to obtain since the bias in estimating entropy has not been addressed. However, their bounds [Loh and Nowozin, 2013, Corollary 2] could be used in place of Theorem 3a. While other work proposes more accurate estimators for entropy [Nowozin, 2012; Paninski, 2003; Schürmann, 2004], they are not computationally efficient and thus not directly useful in our setting.

Finally, greedy maximization is known to be *robust to noise* [Streeter and Golovin, 2009; Krause and Golovin, 2014]: if instead of selecting $i^G = \arg\max_{i \in \mathcal{X} \setminus \mathcal{A}^G} \Delta(i|\mathcal{A}^G)$, we selects i' such that $\Delta(i'|\mathcal{A}^G) \geq \Delta(i^G|\mathcal{A}^G) - \epsilon_1$, the total error is bounded by $\epsilon = k\epsilon_1$. We exploit this property in our method but use confidence bounds to introduce a probabilistic element, such that with high probability $\Delta(i^P|\mathcal{A}^G) \geq \Delta(i^G|\mathcal{A}^G) - \epsilon_1$.

8 Experiments & Results

We evaluated PAC greedy maximization on the problem of tracking multiple people using a multi-camera system. The problem was extracted from a real-world dataset collected in a shopping mall. The dataset was gathered over 4 hours using 13 CCTV cameras located in a shopping mall. Each camera uses a *FPDW* pedestrian detector [Dollár *et al.*, 2010] to detect people in each camera image and *in-camera tracking* [Bouma *et al.*, 2013] to generate tracks of the detected people's movement over time. The dataset thus consists of 9915 trajectories, each specifying one person's x-y position.

To evaluate a given algorithm, a trajectory was sampled randomly. At each timestep in the trajectory, a subset of k cameras out of $n = 20$ were selected by the algorithm. Using the resulting observations, the person was tracked using an unweighted particle filter [Doucet *et al.*, 2001], starting from a random initial belief. At each timestep, a prediction $\arg\max_s b(s)$ about the person's location was compared to the person's true location. Performance is the total number of correct predictions made over multiple trajectories.

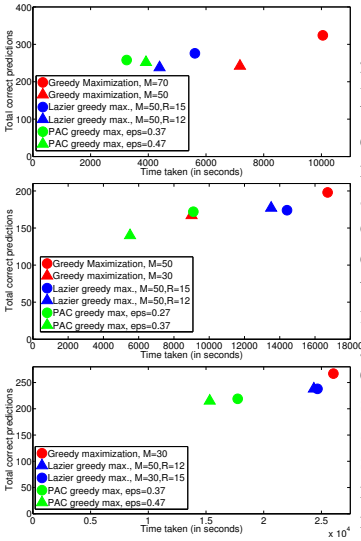

Figure 1: Multi-person tracking for $n = 20$ and (top) $k = 1$; (middle) $k = 2$; (bottom) $k = 3$.

As baselines, we use greedy maximization and lazier greedy maximization. Since we cannot compute F exactly, greedy maximization simply uses an approximation, based on MLE estimates of conditional entropy, ignoring the resulting bias and confidence bounds. Lazier greedy maximization, in each iteration, samples a subset of size R from $\mathcal{X}$ and selects from that subset the element that maximizes the estimated marginal gain. Neither greedy nor lazier greedy maximization employ lazy evaluations because the reliance on approximation of F means pruning is no longer justified. In addition, since lazy greedy maximization's pruning is based on marginal gain instead of F, the bias is exacerbated by the presence of two entropy approximation instead of one. Figure 1 shows the number of correct predictions (y-axis) against the runtime (x-axis) of each method at various settings of M, R and ϵ. Thus, the top left is the most desirable region. In general, PAC greedy maximization performs nearly as well as the best-performing algorithm but does so at lower computational cost. Naively decreasing the number of samples only worsens performance and does not scale with k as the computational cost of even performing greedy maximization with nominal samples is huge in the bottom plot. PAC greedy maximization on the other hand performs consistently in all the three settings and scales much better as k increases, making it more suitable for real-world problems.

References

[Audibert and Bubeck, 2010] Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In *COLT*, 2010.

[Badanidiyuru and Vondrák, 2014] Ashwinkumar Badanidiyuru and Jan Vondrák. *Fast algorithms for maximizing submodular functions*. 2014.

[Bouma *et al.*, 2013] Henri Bouma, Jan Baan, Sander Landsmeer, Chris Kruszynski, Gert van Antwerpen, and Judith Dijk. Real-time tracking and fast retrieval of persons in multiple surveillance cameras of a shopping mall. 2013.

[Bubeck and Cesa-Bianchi, 2012] Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Machine Learning*, 2012.

[Chen and Krause, 2013] Yuxin Chen and Andreas Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In *ICML*, 2013.

[Cover and Thomas, 1991] Thomas M Cover and Joy A Thomas. *Entropy, relative entropy and mutual information*. 1991.

[Dollár *et al.*, 2010] Piotr Dollár, Serge Belongie, and Pietro Perona. The fastest pedestrian detector in the west. In *BMVC*, 2010.

[Doucet *et al.*, 2001] Arnaud Doucet, Nando De Freitas, and Neil Gordon. *Sequential monte carlo methods in practice*. 2001.

[Feder and Merhav, 1994] M. Feder and N. Merhav. Relations between entropy and error probability. 1994.

[Fujishige, 1978] Satoru Fujishige. Polymatroidal dependence structure of a set of random variables. *Information and Control*, 1978.

[Ho *et al.*, 2010] Siu-Wai Ho, T. Chan, and A. Grant. The confidence interval of entropy estimation through a noisy channel. 2010.

[Hoeffding, 1963] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. 1963.

[Joshi and Boyd, 2009] S. Joshi and S. Boyd. Sensor selection via convex optimization. 2009.

[Kovalevskij, 1965] V. A. Kovalevskij. The problem of character recognition from the point of view of mathematical statistics. 1965.

[Krause and Golovin, 2014] Andreas Krause and Daniel Golovin. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*. 2014.

[Krause and Gomes, 2010] Andreas Krause and Ryan G Gomes. Budgeted nonparametric learning from data streams. In *ICML*, 2010.

[Krause and Guestrin, 2005] Andreas Krause and Carlos Guestrin. Near-optimal nonmyopic value of information in graphical models. In *UAI*, 2005.

[Li *et al.*, 2012] Jingxuan Li, Lei Li, and Tao Li. Multi-document summarization via submodularity. 2012.

[Lin and Bilmes, 2010] Hui Lin and Jeff Bilmes. Multi-document summarization via budgeted maximization of submodular functions. 2010.

[Loh and Nowozin, 2013] Po-Ling Loh and Sebastian Nowozin. Faster hoeffding racing: Bernstein races via jackknife estimates. 2013.

[Minoux, 1978] M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. 1978.

[Mirzasoleiman *et al.*, 2013] Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *NIPS*, 2013.

[Mirzasoleiman *et al.*, 2015] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. Lazier than lazy greedy. In *AAAI*, 2015.

[Mow, 1998] W. H. Mow. A tight upper bound on discrete entropy. 1998.

[Nemhauser *et al.*, 1978] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 1978.

[Nowozin, 2012] Sebastian Nowozin. Improved information gain estimates for decision tree induction. In *ICML*, 2012.

[Paninski, 2003] Liam Paninski. Estimation of entropy and mutual information. 2003.

[Radlinski *et al.*, 2008] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. Learning diverse rankings with multi-armed bandits. In *ICML*, 2008.

[Satsangi *et al.*, 2015] Yash Satsangi, Shimon Whiteson, and Frans Oliehoek. Exploiting submodular value functions for faster dynamic sensor selection. In *AAAI*, 2015.

[Schürmann, 2004] Thomas Schürmann. Bias analysis in entropy estimation. 2004.

[Spaan and Lima, 2009] Matthijs T. J. Spaan and Pedro U. Lima. A decision-theoretic approach to dynamic sensor selection in camera networks. In *ICAPS*, 2009.

[Spaan, 2008] Matthijs T. J. Spaan. Cooperative active perception using POMDPs. 2008.

[Streeter and Golovin, 2009] Matthew Streeter and Daniel Golovin. An online algorithm for maximizing submodular functions. In *NIPS*, 2009.

[Valiant, 2013] Leslie Valiant. *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. 2013.

[Wei *et al.*, 2014] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Fast multi-stage submodular maximization. In *ICML*, 2014.

[Williams *et al.*, 2007] J.L. Williams, J.W. Fisher, and A.S. Willsky. Approximate dynamic programming for communication-constrained sensor network management. 2007.