PAC Greedy Maximization with Efficient Bounds on Information Gain for Sensor Selection

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 Liverpool
University of Amsterdam
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 also propose novel, cheap confidence bounds for *conditional entropy*, which appears in many common choices of F and for which it is difficult to find unbipased 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 state can take many values, as it involves an expectation over the entropy of posterior beliefs about the hidden state. 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 ap-

This is a corrected version of this paper. The original version contained a technical mistake in the proof of Lemma 1. We would like to thank Csaba Szepesvári for identifying this mistake.

proximately 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 definitions of F such as information gain. These definitions depend on computing entropy over posterior beliefs, which are impossible to estimate in unbiased way [Paninski, 2003]. The absence of an unbiased estimator renders Hoeffding's inequality inapplicable and 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 $F : 2^{\mathcal{X}} \rightarrow \mathbb{R}$, *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 the highest priority is higher than the priority of the 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.

Assumption 1. We assume access to anytime upper and lower confidence bounds on $F(\mathcal{A})$ and a `tighten`$(\mathcal{A}, t)$ procedure that for all $\mathcal{A} \in \mathcal{A}^+$ that takes in as input arguments $\mathcal{A}$ and t (t is a positive integer) and returns $U_t(\mathcal{A})$ and $L_t(\mathcal{A})$ such that with probability $1 - \frac{\delta_l}{nt(t+1)}$, $L_t(\mathcal{A}) \leq F(\mathcal{A})$ and with probability $1 - \frac{\delta_u}{nt(t+1)}$, $U_t(\mathcal{A}) \geq F(\mathcal{A})$, for some fixed value of δ_l and δ_u. Also, we assume that the lower and upper confidence bounds L_t and U_t are monotonically increasing and decreasing respectively, that is, $L_t \leq L_{t'}$ and $U_t \geq U_{t'}$ for $t' > t$. (Here n is the size of $\mathcal{X} = \{1, 2, \ldots, n\}$ and $\mathcal{A}$ is a subset of $\mathcal{X}$ of size less than or equal to k.)

These assumptions are 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_t and L_t can easily be constructed using, e.g., Hoeffding's inequality. According to Hoeffding's inequality for $x_i \in [0, 1]$,

$$\Pr(|F(\mathcal{A}) - \mathbb{E}[\hat{F}(\mathcal{A})]| \geq \epsilon) \leq 2e^{(-2\epsilon^2 N)}. \tag{2}$$

Using Hoeffding's inequality, L_t and U_t can be constructed as: with probability $1 - \frac{\delta_l}{t(t+1)}$, $L_t(\mathcal{A}) = \hat{F}(\mathcal{A}) - \sqrt{\frac{1}{2N}\log(\frac{2t(t+1)}{\delta_l})} \leq F(\mathcal{A})$ is true and that with probability $1 - \frac{\delta_u}{t(t+1)}$, $U_t(\mathcal{A}) = \hat{F}(\mathcal{A}) + \sqrt{\frac{1}{2N}\log(\frac{2t(t+1)}{\delta_u})} \geq F(\mathcal{A})$. Furthermore, `tighten` procedure can tighten these lower and upper bounds by spending more computation, thereby, using more samples (higher N) to compute $\hat{F}$. However, we specifically do *not* assume access to an unbiased estimator of Q. Instead, we seek an algorithm that performs submodular function maximization given only U_t, L_t, 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 $\mathcal{X} = \{1, 2 \ldots n\}$ 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. $\mathbf{z} = \langle z_1, z_2 \ldots z_n \rangle$ 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:

$$b_{\mathbf{z}}^{\mathcal{A}}(s) = \frac{\Pr(\mathbf{z}|s, \mathcal{A})b(s)}{\Pr(\mathbf{z}|b, \mathcal{A})}, \tag{3}$$

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:

$$H_b^{\mathcal{A}}(s|\mathbf{z}) = \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b,\mathcal{A}) H_{b_{\mathbf{z}}^{\mathcal{A}}}(s), \tag{4}$$

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*:

$$IG_b(\mathcal{A}) = H_b(s) - H_b^{\mathcal{A}}(s|\mathbf{z}). \tag{5}$$

Since the first term in (5) is independent of $\mathcal{A}$, we equivalently define $F(\mathcal{A})$ as:

$$F(\mathcal{A}) = -\sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b,\mathcal{A}) H_{b_{\mathbf{z}}^{\mathcal{A}}}(s). \tag{6}$$

Unfortunately, when there are many possible states and actions, computing $H_{b_{\mathbf{z}}^{\mathcal{A}}}$(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`$(\texttt{tighten}, \mathcal{X}, \mathcal{A}^P, \epsilon_1)$
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 the closely related to best-arm identification algorithms [Audibert and Bubeck, 2010] for multi-armed bandits. Spcifically, 3 is closes to the algorithm Hoeffding's races, presented in [Maron and Moore, 1994; 1997] except that [Maron and Moore, 1994; 1997] propose explicitly to use Hoeffding's inequality to compute and tighten the upper and lower confidence bound[1]. Consequently, the analysis and convergence of the algorithms that they present are reliant on the application of Hoeffding's inequality and, thus, are applicable only for functions that can be estimated in an unbiased manner. This is in contrast to Algorithm 3 and its analysis that is given in the later section, both of which do not necessarily make any assumption on the way in which the upper and lower confidence bounds are generated or tightened.

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]. 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 $F(\mathcal{A}^*)$.

We can now prove a lemma that shows that, with high probability, the marginal gain of the element picked by `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. *Let* $\mathcal{X} = \{1, 2, \ldots, n\}$, $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$, *and* $F : 2^{\mathcal{X}} \rightarrow \mathbb{R}_+$. *If* `pac-max`$(\texttt{tighten}, \mathcal{X}, \mathcal{A}, \epsilon_1)$ *terminates and returns* i^P, *and if Assumption 1 holds, then with probability (at least)* $1 - \delta$,

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

where $\delta_1 = \delta_u + \delta_l$ *and* $i^* = \arg\max_{i\in\mathcal{X}\setminus\mathcal{A}^P} F(\mathcal{A}^P \cup i)$ *and* $\mathcal{A}^P$ *is any set in* $\mathcal{A}^+$.

Proof. If `pac-max` returns $i^P = i^*$, then the Lemma holds trivially, since

$$F(\mathcal{A}^P \cup i^*) \geq F(\mathcal{A}^P \cup i^*) - \epsilon_1. \tag{8}$$

[1]The other *minute* differences between `pac-max` and Hoeffding's races are (a) the use of priority queue in `pac-max`, and (b) that Hoeffding's races do(es) not *explicitly* take into account the number of times `tighten` procedure was previously called as an input parameter to the `tighten` procedure.

Algorithm 3 pac-max($\texttt{tighten}, \mathcal{X}, \mathcal{A}^P, \epsilon_1$)

▷ Input: pac-max takes as input access to tighten procedure; $\mathcal{X} = \{1, 2, \ldots, n\}$ original set of n elements; $\mathcal{A}^P$ a subset of $\mathcal{X}$, in this case, $\mathcal{A}^P$ is the partial solution maintained by pac-greedy-max, ϵ a positive real number.

$i^P \leftarrow 0$ ▷ element with max lower bound
$\rho \leftarrow$ empty priority queue
$t \leftarrow 0$ ▷ t is the iteration number.
$t = t + 1$
for $i \in \mathcal{X} \setminus \mathcal{A}^P$ **do**
  $U_t(i), L_t(i) \leftarrow \texttt{tighten}(\mathcal{A}^P \cup i, t)$

▷ Here $U_t(i)$ and $L_t(i)$ denote the upper and lower bound on $F(\mathcal{A}^P \cup i)$.

  $\rho.\texttt{enqueue}(i, U_t(i))$
  $i^P \leftarrow \arg\max_{j \in \{i, i^P\}} L_t(j)$
end for
while $(\rho.\texttt{length}() > 1)$ **do**
  $\rho' \leftarrow$ empty priority queue
  $t = t + 1$
  i^P-in-queue = True
  ▷ Set flag to check if i^P is still in ρ
  while $\neg\rho.\texttt{empty}()$ **do**
    $i \leftarrow \rho.\texttt{dequeue}()$
    if $i = i^P$ **then**
      i^P-in-queue = False
    end if
    if $(i = i^P) \vee (U_t(i) \geq L_t(i^P) + \epsilon_1)$ **then**
      $U_t(i), L_t(i) \leftarrow \texttt{tighten}(\mathcal{A}^P \cup i, t)$
      $i^P \leftarrow \arg\max_{j \in \{i, i^P\}} L_t(j)$
      $\rho'.\texttt{enqueue}(i, U_t(i))$
    else if i^P-in-queue = True **then**
      Continue
    else
      Break Inner While Loop
    end if
  end while
  $\rho \leftarrow \rho'$
end while
return i^P

For the case that pac-max returns $i^P \neq i^*$, we provide the proof here.

Lets assume that pac-max returns i^P after T (T not known or fixed) total iterations. That is, t goes from 0 to T.

We prove this Lemma in two parts:

- In part A, we show that if the assumed confidence intervals $U_t(i)$ and $L_t(i)$ hold for all t and i, then pac-max returns an ϵ-optimal element. That is, if
$$U_t(i) \geq F(\mathcal{A}^P \cup i) \tag{9}$$
and
$$L_t(i) \leq F(\mathcal{A}^P \cup i) \tag{10}$$
is true for all $i \in \mathcal{X}$ and $t \in \{1, 2, \ldots, T\}$, then
$$F(\mathcal{A}^P \cup i^P) \geq F(\mathcal{A}^P \cup i^*) - \epsilon_1. \tag{11}$$
- In part B, we compute the probability that the confidence intervals hold for all i and t. We show that this probability is lower bounded by $1 - \delta_1$ if Assumption 1 holds. Here $\delta_1 = \delta_l + \delta_u$ is the probability that the confidence intervals (UCI or LCI) are not true at least once in T iterations for at least one i.

Part A: To show, if for all i and t, confidence intervals hold, that is, $U_t(i) \geq F(\mathcal{A}^P \cup i) \geq L_t(i)$ is true for all i, t, then
$$F(\mathcal{A}^P \cup i^P) \geq F(\mathcal{A}^P \cup i^*) - \epsilon_1. \tag{12}$$

At any iteration $t \in \{1, 2, \ldots, T\}$ pac-max maintains the element with max lower bound. Lets denote the element with max lower bound at the end of iteration t by i^P_t. Since i^* was eliminated ($i^P \neq i^*$), thus at some iteration t', its upper bound was lower than the maximum lower bound + ϵ_1 (lets say of element $i^P_{t'}$). Let $L_{t'}(i)$ denote the lower bound (and $U_{t'}(i)$ denote the upper bound) at iteration t' of element i, then, the lower bound of element $i^P_{t'}$ is greater than the upper bound of i^* minus ϵ at some iteration t':
$$L_{t'}(i^P_{t'}) \geq U_{t'}(i^*) - \epsilon_1. \tag{13}$$

Since (a) we have assumed that L_t is monotonically increasing, and (b) pac-max returns i^P, this implies on termination the element with maximum lower bound is i^P, and this lower bound on i^P has to be greater than $L_{t'}(i^P_{t'})$, since i^P was able to replace $i^P_{t'}$ at an iteration $t > t'$.
$$L_T(i^P) \geq L_{t'}(i^P_{t'}) \tag{14}$$

Combining (13), and (14), we get,
$$L_T(i^P) \geq L_{t'}(i^P_{t'}) \geq U_{t'}(i^*) - \epsilon_1 \tag{15}$$

If confidence interval hold for all t and i, then $U_{t'}(i^*) \geq F(\mathcal{A}^P \cup i^*)$ and $F(\mathcal{A}^P \cup i^P) \geq L_T(i^P)$, this implies,
$$F(\mathcal{A}^P \cup i^P) \geq L_T(i^P) \geq U_{t'}(i^*) - \epsilon_1 \geq F(\mathcal{A}^P \cup i^*) - \epsilon_1. \tag{16}$$

Thus, if confidence intervals $U_t(i)$ and $L_t(i)$ hold for all t and i, then,
$$F(\mathcal{A}^P \cup i^P) \geq F(\mathcal{A}^P \cup i^*) - \epsilon. \tag{17}$$

Part B: In part B, we compute the probability that the upper and lower confidence intervals hold for all t and i. The

reasoning in this part follows from the proof presented in [Maron and Moore, 1994; 1997].

We will be using extensively the union bound during this part of the proof. According to union bound the probability of the union of events $A_1, A_2, \ldots, A_l$ is bounded by the sum of their individual probabilities:

$$\Pr(A_1 \vee A_2 \vee \cdots \vee A_l) = \Pr(\bigcup_l A_l) \leq \sum_l \Pr(A_l) \quad (18)$$

To compute the probability that the confidence intervals hold for all i for all t, we observe that this probability is equal to 1 - the probability that the confidence intervals do not hold for at least one i during at least one iteration t. So we want to compute the probability: Pr(upper confidence interval (UCI) OR lower confidence interval (LCI) do not hold for at least one i for at least one value of t).

To compute this probability, lets start with the probability of the confidence interval to NOT hold for one particular $i = i'$ at one particular iteration $t = t'$. We have assumed that at iteration t, $\texttt{tighten}(\mathcal{A}^P \cup i, t)$ returns $U_t(i)$ (and $L_t(i)$) such that with probability $1 - \frac{\delta_u}{nt(t+1)}, U_t(i) \geq F(\mathcal{A}^P \cup i)$ (this condition means that upper confidence interval holds) is true. This implies that for a particular $i = i'$ at iteration $t = t'$, the probability of upper confidence interval to not hold is (less than) $\frac{\delta_u}{nt'(t'+1)}$ and the probability that lower confidence interval ($L_t(i) \leq F(\mathcal{A}^P \cup i)$) does not hold is (less than) $\frac{\delta_l}{nt'(t'+1)}$.

By Assumption 1,

$$\Pr(\text{UCI is not true for } i' \text{ at iteration } t') \leq \frac{\delta_u}{nt'(t'+1)}, \quad (19)$$

and

$$\Pr(\text{LCI is not true for } i' \text{ at iteration } t') \leq \frac{\delta_l}{nt'(t'+1)}, \quad (20)$$

Thus, using union bound,

$$\Pr\Big(\text{UCI OR LCI is not true for } i' \text{ at iteration } t' \Big) \leq \frac{\delta_u + \delta_l}{nt'(t'+1)}. \quad (21)$$

Again using union bound, probability that confidence intervals do NOT hold for i' at $t = 1$ OR $t = 2$ OR $t = 3$ OR $\ldots$ OR $t = T$ is bounded by sum of individual (probability that confidence intervals do NOT hold for i' for $t = 1$) + (probability that confidence intervals do NOT hold for i' for $t = 2$) + $\ldots$ (series ends at $t = T$).

From equation (21), we know the probability that confidence intervals do not hold for i' at iteration t' is less than $\frac{\delta_u+\delta_l}{nt'(t'+1)}$.

$$\Pr((\text{UCI or LCI is not true for } i' \text{ at least once in } t \in \{1, 2, \ldots, T\}) \leq \sum_{t=1}^{T} \frac{\delta_l + \delta_u}{nt(t+1)} \quad (22)$$

The sum over t of the series $\frac{1}{t(t+1)}$, that is $S_T = \sum_{t=1}^{T} \frac{1}{t(t+1)}$ is bounded by $(1 - \frac{1}{T+1})$ for a finite T and even as $\lim T \to \infty$, $\lim_{T\to\infty} \sum_{t=1}^{T} \frac{1}{t(t+1)}$ is bounded by 1[2].

Thus, using union bound,

$$\Pr((\text{UCI or LCI is not true for } i' \text{ at least once in } t \in \{1, 2, \ldots, T\}) \leq \frac{\delta_u + \delta_l}{n} \quad (26)$$

Again we can use union bound to show that the probability that the confidence intervals do not hold for $i = 1$ OR $i = 2$ OR $\ldots$ OR $i = n$, at least once in $t \in \{1, 2, \ldots, T\}$ is bounded by the (probability that the confidence interval do not hold for $i = 1$ at least once in $t \in \{1, 2, \ldots, T\}$) + (probability that the confidence interval do not hold for $i = 2$ at least once in $t \in \{1, 2, \ldots, T\}$) + $\ldots$ + (probability that the confidence interval do not hold for $i = n$ at least once in $t \in \{1, 2, \ldots, T\}$) .

Since for each i the probability that the confidence interval do not hold for i at least once in $t \in \{1, 2, \ldots, T\}$ is bounded by $\frac{\delta_u+\delta_l}{n}$. Taking the sum over n terms yields:

$$\Pr(\text{UCI or LCI is not true for at least one } i \text{ at least once in } t \in \{1, 2, \ldots, T\}) \leq \delta_u + \delta_l. \quad (27)$$

Finally, since probability that confidence intervals hold for all i for all $t \in \{1, 2, \ldots, T\}$ = 1 - probability confidence intervals do not hold at least for one $t \in \{1, 2, \ldots, T\}$ for at least one i, we can write that with probability $1 - \delta_1$,

$$F(\mathcal{A}^P \cup i^P) \geq F(\mathcal{A}^P \cup i^*) - \epsilon_1. \quad (28)$$

□

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

[2] $\sum_{t=1}^{T} \frac{1}{t(t+1)}$ can be expressed as:

$$\sum_{t=1}^{T} \frac{1}{t(t+1)} = \sum_{t=1}^{T} [\frac{1}{t} - \frac{1}{t+1}] \quad (23)$$

$$= [1 - \frac{1}{2} + \frac{1}{2} - \frac{1}{3} + \frac{1}{3} - \cdots - \frac{1}{T+1}] \quad (24)$$

$$= [1 - \frac{1}{T+1}]. \quad (25)$$

by approximating the entropy over a posterior belief. This entropy 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, $\{s^1, s^2 \dots s^M\}$ from a discrete distribution $b(s)$, the *maximum likelihood estimator* (MLE) of $b(s)$ is:

$$\hat{b}(s) = \frac{1}{M} \sum_{j=1}^{M} \mathbb{1}(s^j, s), \tag{29}$$

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:

$$H_{\hat{b}}(s) = \sum_{s} \hat{b}(s) \log(\hat{b}(s)). \tag{30}$$

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

Theorem 2. *(Paninski 2003)*

$$(a)\ \Pr(|H_{\hat{b}}(s) - \mathbb{E}[H_{\hat{b}}(s) \mid b]| \geq \eta) \leq \delta_\eta, \tag{31}$$

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

$$(b)\ \mu_M(b) \leq \mathbb{E}[H_{\hat{b}}(s) \mid b] - H_b(s) \leq 0, \tag{32}$$

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

Hence (31) bounds the variance of $H_{\hat{b}}(s)$ and (32) bounds its bias, which is always negative.

6.1 Lower Confidence Bound

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

$$H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z}) = \sum_{\mathbf{z}_i \in \Omega} \Pr(\mathbf{z}_i|b, \mathcal{A}) H_{\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}}(s), \tag{33}$$

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

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

$$H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z}) \leq H^{\mathcal{A}}_{b}(s|\mathbf{z}) + \eta, \tag{34}$$

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

Proof. For a given $\mathbf{z}_i$, (31) and (32) imply that, with probability $1 - \delta_\eta$,

$$H_{\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}}(s) \leq H_{b_{\mathbf{z}_i}^{\mathcal{A}}}(s) + \eta \tag{35}$$

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 $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z})$ is performing the belief update to find $\hat{b}^{\mathcal{A}}_{\mathbf{z}_i}$ for each $\mathbf{z}_i$. In practice, we approximate these using *particle belief updates* [Doucet *et al.*, 2013], 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, $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z})$ 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 $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z})$ 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 $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z})$.

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 to which z_i belongs. 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. $H^{\mathcal{A}}_{b}(s|\mathbf{z}) \leq H^{\mathcal{A}}_{b}(s|\mathbf{r})$.

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

$$H^{\mathcal{A}}_{b}(s|\mathbf{z}, \mathbf{r}) + H^{\mathcal{A}}_{b}(\mathbf{z}|\mathbf{r}) = H^{\mathcal{A}}_{b}(\mathbf{z}|s, \mathbf{r}) + H^{\mathcal{A}}_{b}(s|\mathbf{r}). \tag{36}$$

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

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

Note that computing $H^{\mathcal{A}}_{b}(s|\mathbf{r})$ requires $\Pr(\mathbf{r}|s, \mathcal{A})$, which can be obtained 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:

$$H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r}) = \sum_{\mathbf{r}_i \in \Phi} \Pr(\mathbf{r}_i|b, \mathcal{A}) H_{\hat{b}_{\mathbf{r}_i}^{\mathcal{A}}}(s). \tag{37}$$

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

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

$$H^{\mathcal{A}}_{b}(s|\mathbf{z}) \leq H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r}) + \eta - \mu_M(b), \tag{38}$$

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

Proof. (32) implies that, for any fixed $\mathbf{r}_i \in \Phi$,

$$H_{b^{\mathcal{A}}_{\mathbf{r}_i}}(s) \leq \mathbb{E}[H_{\hat{b}^{\mathcal{A}}_{\mathbf{r}_i}}(s) \mid b^{\mathcal{A}}_{\mathbf{r}_i}] - \mu_M(b). \tag{39}$$

Taking an expectation on both sides:

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

Now, (31) implies that, with probability $1-\delta_\eta$,

$$\mathbb{E}[H_{\hat{b}^{\mathcal{A}}_{\mathbf{r}_i}}(s) \mid b^{\mathcal{A}}_{\mathbf{r}_i}] \leq H_{\hat{b}^{\mathcal{A}}_{\mathbf{r}_i}}(s) + \eta. \tag{40}$$

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

$$H^{\mathcal{A}}_{b}(s|\mathbf{r}) \leq H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r}) + \eta - \mu_M(b). \qquad \square$$

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

Thus, when computing $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z})$, we set M low but perform many belief updates; when computing $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r})$ 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 3. *Let* $F(\mathcal{A}) = H_b(s) - H^{\mathcal{A}}_{b}(s|\mathbf{z})$. *Let* `tighten` *be defined such that* $\texttt{tighten}(\mathcal{A}, \texttt{t})$ *returns*

$U_t(\mathcal{A}) = H_b(s) - H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z}) + \sqrt{\frac{2(\log(M))^2}{M}\log(\frac{2n|\Omega|t(t+1)}{\delta_u})}$ *and*

$L_t(\mathcal{A}) = H_b(s) - [H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r}) + \sqrt{\frac{2(\log(M))^2}{M}\log(\frac{2n|\Omega|t(t+1)}{\delta_l})} + \log(1+\frac{1}{M}(|supp(b)|-1))]$.

Let $\mathcal{A}^P = \texttt{pac-greedy-max}(\texttt{tighten}, \mathcal{X}, k, \epsilon_1)$ *and* $\mathcal{A}^* = \arg\max_{\mathcal{A}\in\mathcal{A}^+} F(\mathcal{A})$, *where* $\mathcal{X} = \{1, 2, \ldots, n\}$ *and* $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$. *If* $\mathbf{z}$ *is conditionally independent given* s *then, with probability* $1-\delta$,

$$F(\mathcal{A}^P) \geq (1-e^{-1})F(\mathcal{A}^*) - \epsilon, \tag{41}$$

where $\delta = k(\delta_l + \delta_u)$, $\epsilon = k\epsilon_1$.

Proof. We showed that with probability $1-\frac{\delta_l}{nt(t+1)}$, $L_t(\mathcal{A}) \leq F(\mathcal{A})$ and with probability $1-\frac{\delta_u}{tn(t+1)}$, $U_t(\mathcal{A}) \geq F(\mathcal{A})$. Krause and Guestrin, 2005 showed that Q is non-negative, monotone and submodular if $\mathbf{z}$ is conditionally independent given s. The `tighten` procedure can be designed by tightening the upper and lower bounds by either increasing M or by changing the clusters used to estimate $H_{\hat{b}}(s|\mathbf{r})$. Thus, Theorem 12 with $\epsilon = k\epsilon_1$ and $\delta_1 = \delta_u + \delta_l$ implies the stated result. $\square$

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 (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 they can compute the exact posterior entropies; we assume 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 2a. 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 Experimental 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 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. The field of view of a few cameras were divided into two or three separate regions and each region was treated as an independent camera, so as to enable more challenging experiments with as many as $n = 20$ cameras.

We first consider tracking a single person. The hidden state s is modeled as the position and velocity of the person and described by the tuple $\langle x, y, v_x, v_y \rangle$, where x and y describe the position and v_x and v_y describe his velocity in the x and y directions. Both x and y are integers in $\{0, \ldots, 150\}$. The surveillance area can be observed with $n = 20$ cameras and, if selected, each camera produces an observation $\langle z^x, z^y \rangle$ containing an estimate of the person's x-y position.

We assume a person's motion in the x direction is independent of his motion in the y direction. Given the current position x_{curr}, the future position x_{next} is a deterministic function of x_{curr} and the current velocity in x-direction v_x^{curr}, i.e., $x_{next} = x_{curr} + v_x^{curr}$. The same is true for the y position. The future velocity v_{next} is modeled as a Gaussian distribution with the current velocity as the mean and the standard deviation, which depends on the current x-y position, learnt from the data, i.e., $v_x^{next} \sim \mathcal{N}(v_x^{curr}, \sigma^x)$ and $v_y^{next} \sim \mathcal{N}(v_y^{curr}, \sigma^y)$. The observations are assumed to be conditionally independent given the state and are generated from a Gaussian distribution with the true position as the mean and a randomly generated standard deviation. Since ground truth data about people's locations is not available, learning the standard deviation is not possible. A belief $b(s)$ about the person's location was maintained using an unweighted particle filter with 200 particles. Given a subset of the sensors and the observations they generated, $b(s)$ is updated using a particle belief update [Doucet *et al.*, 2013]

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.*, 2013], 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. For multi-person tracking, the best subsets of cameras for each person were computed independently of each other and then the subset with the highest value of F was selected.

We conducted experiments with different values of n and k. As a baseline, 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 making no attempt to reason about 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, i.e., pruning elements via a priority queue as in lazy greedy maximization, 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.

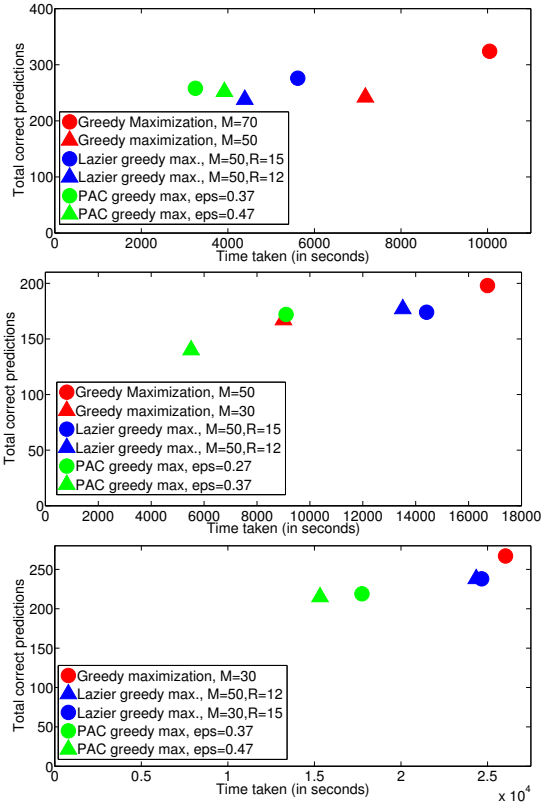

Figure 1: Multi-person tracking for $n = 20$ and (top) $k = 1$; (middle) $k = 2$; (bottom) $k = 3$.

For greedy maximization and lazier greedy maximization, the number of samples M used to approximately compute F was varied from 10 to 100. For lazier greedy maximization, the value of R was also varied from 5 to 15. For PAC greedy maximization, the number of samples used to compute the upper and lower bounds were fixed to 20 and 10 respectively, while the parameter ϵ_1 was varied from 0.1 to 0.5. On average the length of each trajectory sampled was 30 timesteps and the experiments were performed on 30 trajectories for $k = 1$, 17 trajectories for $k = 2$ and 20 trajectories for $k = 3$, with 5 independent runs for $k = 3$ and 3 independent runs for $k = 2$ and $k = 1$. To avoid clutter, we show results for only the two best performing parameter settings of each algorithm.

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 desir-

able 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 well in all the three settings and scales much better as k increases, making it more suitable for real-world problems.

9 Conclusions & Future Work

This paper proposed PAC greedy maximization, a new algorithm for maximizing a submodular function F when computing F exactly is prohibitively expensive. Our method assumes access to cheap confidence bounds on F and uses them to prune elements on each iteration. When F involves entropy, as is common in many applications, obtaining confidence bounds is complicated by the fact that no unbiased estimator of entropy exists. Therefore, we also proposed novel, cheap confidence bounds on conditional entropy that are suitable for use by PAC greedy maximization. We proved that the resulting method has bounded error with high probability. Our empirical results demonstrated that our approach performs comparably to greedy and lazier greedy maximization, but at a fraction of the computational cost, leading to much better scalability. In future work, we aim to develop good strategies for clustering observations to obtain tight upper confidence bounds on conditional entropy, and to combine our upper and lower confidence bounds with more sophisticated best-arm identification algorithm sto produce an even more efficient version of PAC greedy maximization.

Acknowledgments

We thank Henri Bouma and TNO for providing us with the dataset used in our experiments. We also thank the STW User Committee for its advice regarding active perception for multi-camera tracking systems. This research is supported by the Dutch Technology Foundation STW (project #12622), which is part of the Netherlands Organisation for Scientific Research (NWO), and which is partly funded by the Ministry of Economic Affairs. Frans Oliehoek is funded by NWO Innovational Research Incentives Scheme Veni #639.021.336.

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. In *ICML*, 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. 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*. Wiley-Interscience, 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.*, 2013] Arnaud Doucet, Nando De Freitas, and Neil Gordon. *Sequential monte carlo methods in practice*. Springer Science & Business Media, 2013.

[Feder and Merhav, 1994] M. Feder and N. Merhav. Relations between entropy and error probability. 40, 1994.

[Fujishige, 1978] Satoru Fujishige. Polymatroidal dependence structure of a set of random variables. 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. Cambridge University Press, 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. In *NAACL*, 2010.

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

[Maron and Moore, 1994] O. Maron and Andrew Moore. Hoeffding races: Accelerating model selection search for classification and function approximation. In *Advances in Neural Information Processing Systems*, January 1994.

[Maron and Moore, 1997] Oded Maron and Andrew W Moore. The racing algorithm: Model selection for lazy learners. pages 193–225, 1997.

[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*, 14, 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. *Neural computation*, 15(6), 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*. Basic Books, 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.