Second, we prove an upper bound on the expected cumulative regret of our algorithm. The bound is polylogarithmic in time and quadratic in the number of features, as in GLM bandits (Filippi et al. 2010). It also reflects the shape of the optimized submodular function, as in Gabillon *et al.* (2013). In summary, we bring together the concepts of bandits and submodularity in a natural way, which allows to analyze the resulting problem.

Finally, we evaluate our solution on two real-world problems. The first problem is learning of an adaptive policy for movie recommendation. The second problem is learning of an adaptive face detector. In both domains, we demonstrate that high-quality policies can be learned fast, from just several hundred interactions with the environment.

2 Adaptive Submodularity and Learning

Adaptive submodularity (Golovin and Krause 2011) can be viewed as a special form of a *partially-observable Markov decision process (POMDP)* (Sondik 1971). The state of this process ϕ is initially unknown and is revealed by actions A. The goal is to maximize a submodular function $f(A, \phi)$ of the state ϕ and actions A.

Formally, we maximize a function of the form:

$$f : 2^I \times \{-1, 1\}^L \to \mathbb{R}, \tag{1}$$

where $I = \{1, \ldots, L\}$ is a set of L *items* and 2^I is its power set. The first argument of f is a set of *chosen items* $A \subseteq I$. The second argument is the *state* $\phi \in \{-1, 1\}^L$ of all items. The i-th entry of ϕ, $\phi[i]$, is the state of item i. The state ϕ is drawn i.i.d. from some probability distribution $P(\Phi)$. The return for choosing items A in state ϕ is $f(A, \phi)$. Without loss of generality, we assume that $f(\emptyset, \phi) = 0$ in all states ϕ. The state is only partially observed. An observation is a vector $\mathbf{y} \in \{-1, 0, 1\}^L$ such that its non-zero entries are the observed states of items. More specifically, we say that $\mathbf{y}$ is an *observation* of state ϕ, and write $\phi \sim \mathbf{y}$, if $\mathbf{y}[i] = \phi[i]$ in all non-zero entries of $\mathbf{y}$. The state ϕ can be also viewed as a *realization* of $\mathbf{y}$. We denote by $\mathrm{dom}(\mathbf{y}) = \{i : \mathbf{y}[i] \neq 0\}$ the set of *observed items* in $\mathbf{y}$ and by $\phi\langle A \rangle$ the observation of items A in state ϕ. We assume that the observations are partially ordered and write $\mathbf{y}' \succeq \mathbf{y}$ if $\mathbf{y}'[i] = \mathbf{y}[i]$ in all non-zero entries of $\mathbf{y}$, $\mathbf{y}'$ is a more specific observation than $\mathbf{y}$.

We illustrate our notation on a simple example. Let $\phi = (1, 1, -1)$ be a state, and $\mathbf{y}_1 = (1, 0, 0)$ and $\mathbf{y}_2 = (1, 0, -1)$ be observations. Then all of the following claims are true:

$$\begin{array}{lll} \phi \sim \mathbf{y}_1, & \mathbf{y}_2 \succeq \mathbf{y}_1, & \phi\langle\{1,3\}\rangle = \mathbf{y}_2, \\ \phi \sim \mathbf{y}_2, & \mathrm{dom}(\mathbf{y}_2) = \{1, 3\}, & \phi\langle\mathrm{dom}(\mathbf{y}_1)\rangle = \mathbf{y}_1. \end{array}$$

Our goal is to maximize the expected value of f by adaptively choosing K items. The items are chosen sequentially according to some policy π and we observe the state of each chosen item. A *policy*:

$$\pi : \{-1, 0, 1\}^L \to I \tag{2}$$

is a function from observations $\mathbf{y}$ to items. The observation represents our past decisions and their outcomes. A *k-step policy* in state ϕ, $\pi_k(\phi)$, is the collection of the first k items chosen by policy π. The policy is defined recursively as:

$$\begin{aligned} \pi_k(\phi) &= \pi_{k-1}(\phi) \cup \left\{\pi_{[k]}(\phi)\right\} \\ \pi_{[k]}(\phi) &= \pi(\phi\langle\pi_{k-1}(\phi)\rangle) \\ \pi_0(\phi) &= \emptyset, \end{aligned} \tag{3}$$

where $\pi_{[k]}(\phi)$ is the k-th item chosen by policy π in state ϕ. The optimal K-step policy satisfies:

$$\pi^* = \arg\max_\pi \mathbb{E}_\phi[f(\pi_K(\phi), \phi)]\,. \tag{4}$$

It is NP-hard to compute the policy π^* (Golovin and Krause 2011). However, near-optimal policies can be computed efficiently when the optimized function has a *diminishing return* property. Specifically, we require that the function f is adaptive submodular and monotonic.

Definition 1. *Function f is* adaptive submodular *if:*

$$\begin{aligned} &\mathbb{E}_\phi[\, f(A \cup \{i\}, \phi) - f(A, \phi) \,|\, \phi \sim \mathbf{y}_A \,] \\ &\quad \geq \mathbb{E}_\phi[\, f(B \cup \{i\}, \phi) - f(B, \phi) \,|\, \phi \sim \mathbf{y}_B \,] \end{aligned}$$

for all items $i \in I \setminus B$ and observations $\mathbf{y}_B \succeq \mathbf{y}_A$ such that $A = \mathrm{dom}(\mathbf{y}_A)$ and $B = \mathrm{dom}(\mathbf{y}_B)$.

Definition 2. *Function f is* adaptive monotonic *if:*

$$\mathbb{E}_\phi[\, f(A \cup \{i\}, \phi) - f(A, \phi) \,|\, \phi \sim \mathbf{y}_A \,] \geq 0$$

for all items $i \in I \setminus A$ and observations $\mathbf{y}_A$ such that $A = \mathrm{dom}(\mathbf{y}_A)$.

In other words, the expected gain of choosing items is non-negative and does not increase as the observations become more specific. Let:

$$\pi^g(\mathbf{y}) = \operatorname*{arg\,max}_{i \in I \setminus \mathrm{dom}(\mathbf{y})} g_i(\mathbf{y}) \tag{5}$$

be the *greedy policy* for maximizing f and:

$$\begin{aligned} g_i(\mathbf{y}) = \mathbb{E}_\phi[&f(\mathrm{dom}(\mathbf{y}) \cup \{i\}, \phi) - \\ &f(\mathrm{dom}(\mathbf{y}), \phi) \mid \phi \sim \mathbf{y}] \end{aligned} \tag{6}$$

denote the *expected gain* of choosing item i after observing $\mathbf{y}$. Then π^g is a $(1 - 1/e)$-approximation to π^*:

$$\mathbb{E}_\phi[f(\pi^g_K(\phi), \phi)] \geq (1 - 1/e)\mathbb{E}_\phi[f(\pi^*_K(\phi), \phi)] \tag{7}$$

when the function f is adaptive submodular and monotonic (Golovin and Krause 2011).

The greedy policy π^g cannot be computed when the distribution $P(\Phi)$ is unknown. Gabillon *et al.* (2013) proposed a bandit algorithm for such problems, *Optimistic Adaptive Submodular Maximization (*`OASM`*)*, that learns $P(\Phi)$ by interacting with the environment. A major assumption in this work is that $P(\Phi)$ is a product of L Bernoulli distributions, one for the state of each item. In this setting, the problem of learning $P(\Phi)$ reduces to learning L parameters, one mean p_i for each distribution. Unfortunately, the expected cumulative regret of `OASM` grows linearly with L. As a result, the method is impractical when L is large.

3 Optimistic Adaptive Submodularity with Generalized Linear Models

In this work, we propose a learning algorithm whose regret does not depend on the number of items L. We assume that each item is associated with a feature vector $\mathbf{x}_i$ and that the states of the item are distributed conditionally on $\mathbf{x}_i$.

3.1 Model

The *feature vector* of item i is a d-dimensional column vector $\mathbf{x}_i \in \mathbb{R}^d$ such that $\|\mathbf{x}_i\|_2 \leq c$. To simplify our notation, we define $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_L) \in \mathbb{R}^{dL}$, a vector of all feature vectors. We assume that the vector $\mathbf{x}$ is known and does not change with time t.

The distribution of states is factored as:

$$P(\Phi = \phi) = \prod_{i=1}^{L} p_i(\mathbf{x})^{\mathbb{1}\{\phi[i]=1\}} \left(1 - p_i(\mathbf{x})\right)^{\mathbb{1}\{\phi[i]=-1\}}, \quad (8)$$

where $p_i(\mathbf{x})$ is the probability that item i is in state 1. The probability $p_i(\mathbf{x})$ is unknown and we want to learn it. Since the state of each item i is binary, the dependency on $\mathbf{x}_i$ can be naturally modeled using *logistic regression*:

$$p_i(\mathbf{x}) = P(\phi[i] = 1 \mid \mathbf{x}_i) = \mu(\mathbf{x}_i^\top \theta^*), \quad (9)$$

where $\mu(u) = 1/(1 + e^{-u})$ is a logistic function, $\theta^* \in \Theta$ is a column vector of learned parameters such that $\|\theta^*\|_2 \leq S$, and $\Theta \subset \mathbb{R}^d$ is a compact and convex space.

Our choice of μ is motivated by the fact that the states of all items in our problem are binary. However, note that this assumption is only for simplicity of exposition. Our results apply to any generalized linear model. In fact, we only need to assume that μ is z_μ-Lipschitz continuous, increasing, and continuously differentiable such that $c_\mu = \inf_{\theta \in \Theta, i} \dot{\mu}(\mathbf{x}_i^\top \theta) > 0$.

With our model in mind, we rewrite the expected gain of choosing item i (Equation 6) as:

$$g_i(\mathbf{y}) = p_i(\mathbf{x}) \bar{g}_i(\mathbf{y}), \quad (10)$$

where:

$$\bar{g}_i(\mathbf{y}) = \mathbb{E}_\phi[f(\mathrm{dom}(\mathbf{y}) \cup \{i\}, \phi) - f(\mathrm{dom}(\mathbf{y}), \phi) \mid \phi \sim \mathbf{y}, \phi[i] = 1] \quad (11)$$

is the expected gain of choosing item i in state 1.[1] In many problems of interest, the gain $\bar{g}_i(\mathbf{y})$ is only a function of $\mathbf{y}$ and i (Gabillon et al. 2013), and therefore can be computed without knowing $P(\Phi)$.

Our learning problem is episodic. In episode t, we adaptively select K items according to some policy π^t, which is episode dependent. The state in episode t is ϕ_t. The quality of π^t is measured by its *expected cumulative K-step return*:

$$\mathbb{E}_{\phi_1, \ldots, \phi_n} \left[\sum_{t=1}^{n} f(\pi_K^t(\phi_t), \phi_t) \right]. \quad (12)$$

[1] For simplicity of exposition, we assume that the expected gain of choosing item i in state -1 is zero. Our work can be straightforwardly generalized to the problems where this gain is non-zero.

The difference between the expected returns of π^g and π^t is the *expected cumulative regret* of π^t:

$$R(n) = \mathbb{E}_{\phi_1, \ldots, \phi_n} \left[\sum_{t=1}^{n} R_t(\phi_t) \right], \quad (13)$$

where $R_t(\phi_t) = f\left(\pi_K^g(\phi_t), \phi_t\right) - f(\pi_K^t(\phi_t), \phi_t)$. In maximum coverage problems, the policy π^g is a good surrogate for π^* because it is a $(1 - 1/e)$-approximation to π^*.

3.2 Algorithm

Our learning algorithm has two main parts. First, at the beginning of each episode, we estimate θ^* from past observation using regression. Second, in each step of each episode, we behave optimistically and greedily choose the item with the highest upper confidence bound on its gain. More precisely, at step k of episode t, we select item $\pi_{[k]}^t(\phi_t)$, which we abbreviate as $\pi_{[k]}^t$, and then observe its state $\phi_t[\pi_{[k]}^t]$.

The pseudo-code of our method is given in Algorithm 1. The parameters θ^* are estimated as follows. First, we compute the maximum-likelihood estimate $\hat{\theta}_t$ of θ^*. By definition, $\hat{\theta}_t$ maximizes the regularized log-likelihood:

$$\begin{aligned}
&\sum_{s=1}^{t-1} \sum_{k=1}^{K} \left[\mathbb{1}\left\{\phi_s[\pi_{[k]}^s] = 1\right\} \log(p_{\pi_{[k]}^s}(\mathbf{x})) + \right. \\
&\qquad \left. \mathbb{1}\left\{\phi_s[\pi_{[k]}^s] = -1\right\} \log(1 - p_{\pi_{[k]}^s}(\mathbf{x})) \right] - \frac{1}{2}\lambda \|\theta\|_2^2 \\
&= \sum_{s=1}^{t-1} \sum_{k=1}^{K} \left[\mathbb{1}\left\{\phi_s[\pi_{[k]}^s] = 1\right\} \log\left(\frac{p_{\pi_{[k]}^s}(\mathbf{x})}{1 - p_{\pi_{[k]}^s}(\mathbf{x})} \right) + \right. \\
&\qquad \left. \log(1 - p_{\pi_{[k]}^s}(\mathbf{x})) \right] - \frac{1}{2}\lambda \|\theta\|_2^2 \\
&= \sum_{s=1}^{t-1} \sum_{k=1}^{K} \left[\mathbb{1}\left\{\phi_s[\pi_{[k]}^s] = 1\right\} \mathbf{x}_{\pi_{[k]}^s}^\top \theta - \right. \\
&\qquad \left. \log\left(1 + e^{\mathbf{x}_{\pi_{[k]}^s}^\top \theta}\right) \right] - \frac{1}{2}\lambda \|\theta\|_2^2, \quad (14)
\end{aligned}$$

which is concave in θ. Upon differentiating, we obtain that $\hat{\theta}_t$ is a unique solution to:

$$\sum_{s=1}^{t-1} \sum_{k=1}^{K} \left[\mathbb{1}\left\{\phi_s[\pi_{[k]}^s] = 1\right\} - \mu\left(\mathbf{x}_{\pi_{[k]}^s}^\top \theta\right) \right] \mathbf{x}_{\pi_{[k]}^s} - \lambda\theta = 0. \quad (15)$$

Second, to solve a technical detail in our proof, we need to guarantee that our solution is in Θ. Therefore, we project $\hat{\theta}_t$ into Θ. For this purpose, we define an invertible function:

$$h_t : \theta \mapsto \sum_{s=1}^{t-1} \sum_{k=1}^{K} \mu\left(\mathbf{x}_{\pi_{[k]}^s}^\top \theta\right) \mathbf{x}_{\pi_{[k]}^s} + \lambda\theta \quad (16)$$

and let:

$$\tilde{\theta}_t = \arg\min_{\theta \in \Theta} \left\| h_t(\theta) - h_t(\hat{\theta}_t) \right\|_{M_t^{-1}} \quad (17)$$

be the resulting projection, where:

$$M_t = \sum_{s=1}^{t-1} \sum_{k=1}^{K} \mathbf{x}_{\pi^s_{[k]}} \mathbf{x}_{\pi^s_{[k]}}^\intercal + \lambda I \tag{18}$$

is the Gram matrix of all feature vectors $\mathbf{x}_{\pi^s_{[k]}}$ up to episode t and I is a $d \times d$ identity matrix. We add the regularization term λI to guarantee that M_t is invertible. The influence of this term vanishes as the number of episodes increases.

At each step k, we compute the *index* of each item i:

$$\left[\mu(\mathbf{x}_i^\intercal \tilde{\theta}_t) + \beta^i_{k,t}(\delta)\right] \bar{g}_i(\mathbf{y}). \tag{19}$$

The term:

$$\beta^i_{k,t}(\delta) = \rho_{k,t}(\delta) \|\mathbf{x}_i\|_{M_t^{-1}} \tag{20}$$

is the radius of the confidence interval around $\mu(\mathbf{x}_i^\intercal \tilde{\theta}_t)$, the estimate of the probability that item i is in state 1. The norm is defined as $\|\mathbf{u}\|_{M_t^{-1}} = \sqrt{\mathbf{u}^\intercal M_t^{-1} \mathbf{u}}$. Therefore, $\|\mathbf{x}_i\|_{M_t^{-1}}$ can be interpreted as the variance of our regressor at item i. The quantity $\rho_{k,t}(\delta)$ is a slowly increasing function in t and k, which is defined as:

$$\rho_{k,t}(\delta) = \frac{4 z_\mu}{c_\mu} (3 + 2 \log(1 + 2c^2/\lambda)) \times \left[\sqrt{2d \log(tK + k) \log(2(tK + k)nK/\delta)} + \lambda^{\frac{1}{2}} S\right]. \tag{21}$$

Based on this setting, $\left|\mu(\mathbf{x}_i^\intercal \theta^*) - \mu(\mathbf{x}_i^\intercal \tilde{\theta}_t)\right| \leq \beta^i_{k,t}(\delta)$ with probability of at least $1 - \delta$, simultaneously for all episodes $t \leq n$ and steps.

After we compute all indices, we select the item with the highest upper confidence bound. The indices enforce exploration of items that have not been chosen very often. As the number of past episodes increases, we get a better estimate of θ^*, all confidence intervals $\|\mathbf{x}_i\|_{M_t^{-1}}$ shrink, and we start exploiting best items.

`LinOASM` is a greedy method. Therefore, our policies can be computed very fast. In particular, the time complexity of `LinOASM` at step k is $O(L)$, because we need to recompute the index of each item (Equation 19). Moreover, estimation of $\hat{\theta}_t$ (Equation 15) is straightforward, because this is just a regression problem. Finally, our solutions are guaranteed to converge to π^g. This claim is proved in the next section.

3.3 Analysis

In this section, we prove a gap-dependent bound on the expected cumulative regret of `LinOASM`. Our analysis is based on counting the number of times when the policy π^t selects a different item from the policy π^g at step k. Therefore, we define several variables that allow us to describe the state of our problem at this step. We denote by:

$$\mathcal{Y}_k(\pi) = \bigcup_{\phi} \{\phi \langle \pi_{k-1}(\phi) \rangle\} \tag{22}$$

the set of all possible observations after policy π is applied for $k - 1$ steps. The observations associated with the policy π^g are denoted by $\mathcal{Y}_k = \mathcal{Y}_k(\pi^g)$. We denote by:

$$\mathcal{Y}_{k,i} = \mathcal{Y}_k \cap \{\mathbf{y} : i \neq \pi^g(\mathbf{y})\} \tag{23}$$

Algorithm 1 `LinOASM`: Optimistic adaptive submodularity with a generalized linear model.

Input:
Feature vectors $\mathbf{x}_1, \ldots, \mathbf{x}_L$
Regularization parameter λ

for all $t = 1, \ldots, n$ **do**
  Estimate $\tilde{\theta}_t$ based on Equations 15 and 17
  $A \leftarrow \emptyset$
  for all $k = 1, \ldots, K$ **do**
    $\mathbf{y} \leftarrow \phi_t \langle A \rangle$
    $j \leftarrow \arg\max_{i \in I \setminus A} \left[\mu(\mathbf{x}_i^\intercal \tilde{\theta}_t) + \beta^i_{k,t}(\delta)\right] \bar{g}_i(\mathbf{y})$
    $A \leftarrow A \cup \{j\}$
  end for
end for

the set of observations where item i is suboptimal at step k. The hardness of discriminating items i and $\pi^g(\mathbf{y})$ in context $\mathbf{y}$ is measured by the *gap* between the expected gains of the items:

$$\Delta_i(\mathbf{y}) = g_{\pi^g(\mathbf{y})}(\mathbf{y}) - g_i(\mathbf{y}). \tag{24}$$

Our main result is presented below.

Theorem 1. *Let $\lambda \geq d^2$. Then the expected cumulative regret of* `LinOASM` *is bounded as:*

$$R(n) \leq \underbrace{\sum_{k=1}^{K} G_k(\ell_k + n\delta)}_{O(\log^3 n)},$$

where $G_k = (K - k + 1) \max_{\mathbf{y} \in \mathcal{Y}_k} \max_i g_i(\mathbf{y})$ is an upper bound on the expected gain of π^g from step k forward and:

$$\ell_k = \lceil d/K \rceil + 8d\rho^2_{K,n}(\delta) \times \left[\max_i \max_{\mathbf{y} \in \mathcal{Y}_{k,i}} \frac{\bar{g}_i^2(\mathbf{y})}{\Delta_i^2(\mathbf{y})}\right] \log\left(\frac{c^2 nK}{\lambda}\right)$$

is the number of episodes after which no item is likely to be chosen suboptimally at step k.

Proof. The proof of our claim is in Appendix. It consists of three main parts. First, we associate the regret in episode t with the first step where π^t selects a different item from π^g. Second, we bound the expected regret in each episode t by the probability of deviating from the policy π^g at step k and an upper bound on the associated loss G_k, which is derived based on the submodularity of f. This part of the analysis is motivated by the work of Gabillon *et al.* (2013). Finally, we bound the expected number of suboptimally selected items based on a high-probability upper bound on the cumulative regret of our regressor. This part of the proof borrows ideas from Filippi *et al.* (2010). ■

The scalar δ is a parameter of our learning algorithm. For a given number of episodes n, it should be set as $\delta = 1/n$. In

Symbol	Description
n	Number of episodes
K	Number of steps in each episode
d	Number of features
$\mathbf{x}_i$	Feature vector of item i
c	Upper bound on $\|\mathbf{x}_i\|_2$
θ^*	Unknown parameter vector
S	Upper bound on $\|\theta^*\|_2$
Θ	Compact and convex space such that $\theta^* \in \Theta$
μ	Mean function
z_μ	Lipschitz constant of μ
c_μ	Lower bound on $\dot{\mu}$
λ	Regularization parameter
$\hat{\theta}_t$	ML estimate of θ^* in episode t
$\tilde{\theta}_t$	Estimate of θ^* in episode t
M_t	Gram matrix in episode t
$\beta^i_{k,t}$	Confidence radius for $\mu(\mathbf{x}_i^\intercal \tilde{\theta}_t)$
$\rho_{k,t}$	Slowly increasing function in t and k
$1-\delta$	Probability that our regret bound holds

Table 1: Summary of our GLM bandit notation.

this case, our regret bound is polylogarithmic in n, through the terms ℓ_k. This dependency on n is the same as in GLM bandits (Filippi et al. 2010).

Our regret bound also captures the submodular character of our problem. Specifically, because f is adaptive submodular, the upper bound on the expected gain of the policy π^g from step k forward, G_k, decreases as k increases. Consequently, earlier deviations from the policy π^g are penalized more than later ones.

Remarkably, our bound is polynomial in all constants of interest. More specifically, it is linear in K and quadratic in the number of features d. This is a significant improvement over Gabillon *et al.* (2013), whose regret bound is linear in the number of items L. In practice, d is often much smaller than L, and we expect significant speedups in learning over `OASM` (Gabillon et al. 2013).

The regularization parameter λ is problem specific. Note that our bound is linear in λ, through the dependency on the term $\rho^2_{K,n}(\delta)$ (Equation 21).

3.4 Independence Assumption

Our independence assumption (Equation 8) allows us to develop a conceptually simple learning algorithm that can be analyzed. In practice, the assumption may be violated. This issue can be addressed in two ways. First, the ground set I is reduced to the items whose states are distributed independently. This is a domain-specific preprocessing step, which is in a sense equivalent to feature selection. In this case, we can guarantee that `LinOASM` converges to the greedy policy π^g on the reduced set I. Second, `LinOASM` is applied to the problem even if the states of items are not distributed independently. In this case, `LinOASM` converges to some suboptimal solution.

In our experiments, the states of items are not distributed independently. The difference between the expected returns of `LinOASM` and π^g is the loss due to our modeling assumptions, one of which is Equation 8.

4 Experiments

`LinOASM` is evaluated on two real-world problems. The first problem is learning of a policy for movie recommendation. The problem is motivated by a movie rating game on *Netflix* (net 2013), where the users are asked to rate a set of movies to improve their profile. The second problem is learning of an adaptive face detection policy. The problem is motivated by a range of adaptive sensing problems in computer vision (Tarabanis, Allen, and Tsai 1995; Butko and Movellan 2009; Vijayanarasimhan and Kapoor 2010). In all of these problems, the computational resources of the algorithm are limited and as a result the algorithm must behave intelligently. Both of our problems are set up such that we can showcase the benefits of submodularity.

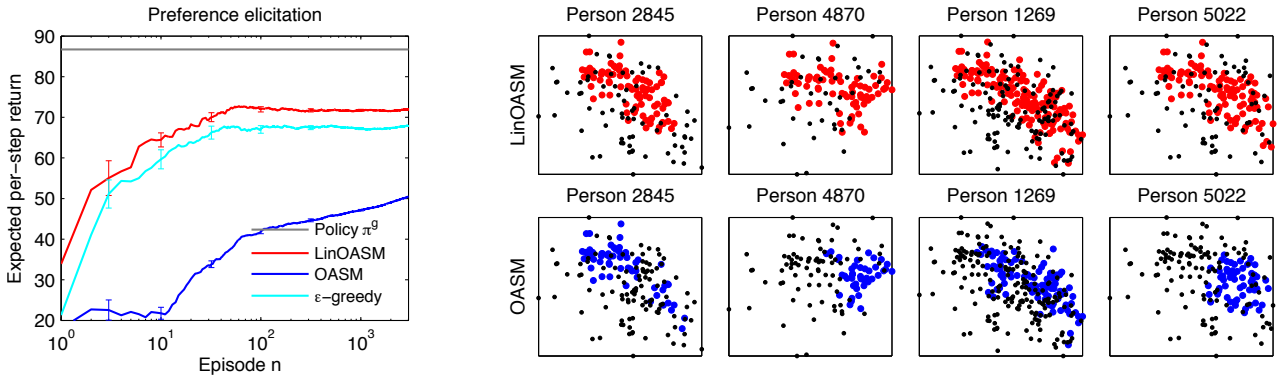
All policies are evaluated by the *expected per-step return* in n episodes, the expected cumulative return in n episodes divided by n. The parameterization of `LinOASM` suggested by our analysis (Section 3.3) is too conservative in practice, as in GLM bandits (Filippi et al. 2010). In our experiments, we choose $\lambda = 1$ and $\rho_{k,t}(\delta) = 0.1 \log(tK + k)$. At these values, `LinOASM` performs well and is not sensitive to small perturbations of the parameters. Note that $\rho_{k,t}(\delta)$ is chosen such that it depends on t and k at the same rate as in Equation 21.

`LinOASM` is compared to three baselines. The first baseline is `OASM` (Gabillon et al. 2013), a learning algorithm for adaptive submodular maximization that does not utilize the features of items. The second baseline is an ε-greedy policy with the same linear generalization model as `LinOASM`. We set ε to 0.1. This method can be viewed as a naive solution to our problem that leverages its structure. Finally, the third baseline is π^g (Equation 5). The policy π^g knows $P(\Phi)$ and does not assume any structure. Therefore, π^g can be viewed as an upper bound on the performance of `LinOASM`.

4.1 Preference Elicitation

In the first experiment, we learn a policy for recommending movies. Our learning problem is episodic. In each episode, we adaptively show K movies to a user. The user may like or dislike each of the movies. Our objective is to maximize the expected number of movies that we can recommend to the user at the end of each episode, movies that are similar to the movies that the user likes.

We experiment with 6k users and 500 most rated movies from the *MovieLens* dataset (Lam and Herlocker 2013). The movie preferences of any user j are represented by a vector $\phi_j \in \{-1, 1\}^L$ such that $\phi_j[i] = 1$ if the user likes movie i and $\phi_j[i] = -1$ otherwise. All movie preference vectors ϕ_j are estimated from historical ratings of the users using low-rank matrix factorization (Koren, Bell, and Volinsky 2009), a standard approach in collaborative filtering. In particular, we factor the user-item rating matrix as $M \approx UX^\intercal$, where U is the user matrix, X is the item matrix, and the number of latent factors is $d = 10$. Based on our model, user j rates movie i as $r_{j,i} = \langle \mathbf{u}_j, \mathbf{x}_i \rangle$, where $\mathbf{u}_j$ and $\mathbf{x}_i$ are the j-th row



(a) **(b)**

Figure 1: a. The expected per-step return of four preference elicitation policies up to episode $n = 3$k. b. Visualization of the `OASM` and `LinOASM` policies in episode $n = 100$. The black dots are liked movies, projected on the first two components of the latent space. The colored dots are liked movies that are similar to at least one queried movie $i \in A$ that is liked.

of U and the i-th row of X, respectively. We set $\phi_j[i]$ to 1 if movie i is rated highly by user j, $r_{j,i} > 4$.

We say that movies i_1 and i_2 are *similar* if they are close in the item space, $\|\mathbf{x}_{i_1} - \mathbf{x}_{i_2}\|_2 \leq 0.2$. We choose this similarity measure because it implies that the ratings of similar movies are similar. More precisely, the $\mathcal{L}_2$-norm of the user factors in our dataset is bounded by 5 and therefore:

$$|r_{j,i_1} - r_{j,i_2}| \leq \|\mathbf{u}_j\|_2 \|\mathbf{x}_{i_1} - \mathbf{x}_{i_2}\|_2 \leq 5 \cdot 0.2. \quad (25)$$

In turn, similar movies are liked similarly.

The return $f(A, \phi)$ for asking a user ϕ questions A is the sum of the returns $f_\ell(A, \phi)$, one for each movie ℓ:

$$f(A, \phi) = \sum_{\ell=1}^{500} f_\ell(A, \phi) \quad (26)$$
$$f_\ell(A, \phi) = 0.1\mathbb{1}\{\exists i \in A : \|\mathbf{x}_\ell - \mathbf{x}_i\|_2 \leq 0.2\} + 0.9\mathbb{1}\{\exists i \in A : \|\mathbf{x}_\ell - \mathbf{x}_i\|_2 \leq 0.2, \phi[i] = 1\}.$$

The return for movie ℓ is 1 if the movie is similar to at least one liked movie i. The return is 0.1 if ℓ is similar to at least one movie i but none of these movies are liked. Otherwise, the return is zero. The function $f(A, \phi)$ is maximized when A is a diverse set of liked movies. It is submodular in A for all ϕ and therefore adaptive submodular when all entries of ϕ are distributed independently.

The features of movie i are $\mathbf{x}_i$. The number of features is $d = 10$. Our goal is to learn how the latent factors correlate with liked movies. We set K to 5.

Our results are reported in Figure 1a. We observe several trends. First, `LinOASM` outperforms `OASM` by a huge margin. In particular, the expected return of `OASM` after 3k episodes is around 50, which is comparable to the return of `LinOASM` in 10 episodes. So `LinOASM` learns more than two orders of magnitude faster than `OASM`. Second, `LinOASM` outperforms the ε-greedy policy. Finally, note that the expected return of `LinOASM` after 3k episodes is much lower than the return of the policy π^g. This gap is due to our modeling assumptions. Specifically, `LinOASM` makes a simplifying assumption that $P(\Phi)$ is factored (Equation 8). This is not true in our problem. As a result, `LinOASM` learns only an approximation of $P(\Phi)$ and is not guaranteed to converge to π^g.

We visualize our solutions in Figure 1b. `LinOASM` clearly covers more liked movies than `OASM`. Therefore, we expect that `LinOASM` would perform significantly better in practice than `OASM`.

4.2 Adaptive Face Detection

In the second experiment, we learn a policy for face detection. Our learning problem is episodic. In each episode, we adaptively sense K regions of an image. Our objective is to maximize the sensed area that contains faces.

We experiment with 3k labeled images of faces from the *GENKI-SZSL* dataset (UCSD 2013). Each image is resized to 100×100 pixels and divided into 81 overlapping *sensing regions*, which are centered at pixels:

$$(u, v) \in \{10, \ldots, 90\} \times \{10, \ldots, 90\}. \quad (27)$$

We assume that all regions are twenty pixels wide and high. Several examples of the regions are in Figure 2b. The state of a region is 1 if at least a quarter of its pixels covers a face and -1 otherwise. Each image in our dataset is represented by a vector $\phi \in \{-1, 1\}^{81}$, which describes the states of all regions in the image.

The return $f(A, \phi)$ for choosing regions A in image ϕ is the average of the returns $f_\ell(A, \phi)$, one for each pixel ℓ:

$$f(A, \phi) = \mathbb{E}_\ell[f_\ell(A, \phi)] \quad (28)$$
$$f_\ell(A, \phi) = 0.1\mathbb{1}\{\exists i \in A : \text{region } i \text{ covers } \ell\} + 0.9\mathbb{1}\{\exists i \in A : \text{region } i \text{ covers } \ell, \phi[i] = 1\}.$$

The return for pixel ℓ is 1 if this pixel is covered by at least one region i with a face. The return is 0.1 if ℓ is covered by at least one region i but none of these regions contain faces. The function $f(A, \phi)$ is maximized when A is a set of non-overlapping regions with faces. It is submodular in A for all states ϕ and therefore adaptive submodular when all entries of ϕ are distributed independently.

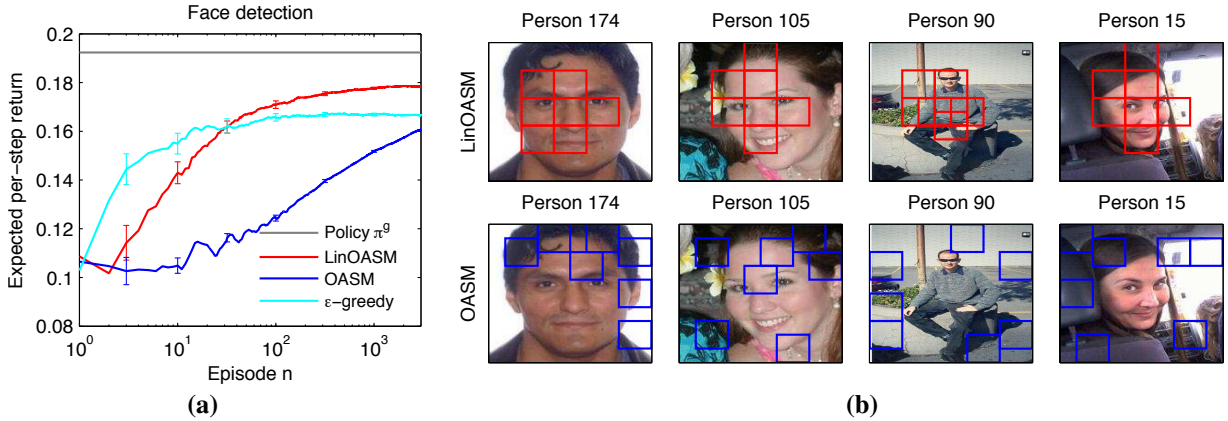


Figure 2: a. The expected per-step return of four face detection policies up to episode $n = 3$k. b. Visualization of the `OASM` and `LinOASM` policies in episode $n = 100$. The colored rectangles are the regions sensed by the policies.

The features of the region summarize the position of the region in the image. Specifically, let region i be centered at (u, v). Then its feature vector is:

$$\mathbf{x}_i = \begin{pmatrix} \|(u,v) - (50,50)\|_2 \\ \max\{u - 50, 0\} \\ \max\{50 - u, 0\} \\ \max\{v - 50, 0\} \\ \max\{50 - v, 0\} \end{pmatrix}. \quad (29)$$

The first feature is the Euclidean distance from the center of the image. The remaining four features are two vertical and two horizontal distances from the center. These features are motivated by the fact that we expect most faces close to the center. We set K to 7.

Our results are reported in Figure 2a. We observe similar trends to those in Section 4.1. Specifically, `LinOASM` learns two orders of magnitude faster than `OASM`. We visualize our solutions in Figure 2b. `LinOASM` clearly covers more facial regions than `OASM`.

5 Related Work

Golovin and Krause (2011) proposed adaptive submodularity, a framework for maximizing adaptive submodular functions. Wen *et al.* (2013) proposed a learning algorithm for a special class of adaptive submodularity, adaptive minimum set cover, where exploration is not necessary. The first bandit algorithm for adaptive submodularity was proposed and analyzed by Gabillon *et al.* (2013). Unfortunately, this solution does not scale to large problems. In this work, we build on the idea of Gabillon *et al.* (2013) and propose a scalable learning algorithm for adaptive submodularity.

The key assumption in our paper is that the state of each item is distributed according to a generalized linear model, which is conditioned on the features of the item. Our generalization model is the same as in GLM bandits (Filippi et al. 2010), and both our approach and its analysis are motivated by recent work on linear bandits, such as Dani *et al.* (2008) and Abbasi-Yadkori *et al.* (2011). Linear bandits were proposed by Aurer (2002) and popularized by Li *et al.* (2010) as a model for personalization.

Our work is also related to Yue and Guestrin (2011). Yue and Guestrin (2011) propose a bandit algorithm that learns how to maximize a submodular function, where the gain of the item is a linear function of its features. The main difference in our work is that our maximization problem is adaptive. The best adaptive policy is often much better than the best non-adaptive policy (Golovin and Krause 2011), but it is also harder to learn. In particular, the adaptive policy is a decision tree while the non-adaptive policy is a set of items. In this paper, we address the harder of the two problems and demonstrate that it can be solved at scale.

6 Conclusions

In this work, we study the problem of learning how to maximize adaptive submodular functions. We assume that each item is associated with a feature vector and that the state of the item is distributed conditionally on its features. We propose a learning algorithm that leverages this structure, and show that its regret is polylogarithmic in time and quadratic in the number of features. Our solution is evaluated on two problems and we demonstrate that high-quality policies can be learned sample efficiently. Submodularity is common in practice and we believe that our method can solve many interesting active learning problems, such as learning policies for near-optimal viral marketing in social networks (Kempe, Kleinberg, and Tardos 2003).

Our method and its analysis can be easily generalized to the setting where the feature vectors change with time (Filippi 2010). This generalization could be of interest in many applications. For instance, the features in our face detection problem (Section 4.2) could then depend on the content of the image and be more informative.

Adaptive submodularity can be viewed as a special form of POMDPs (Section 2). Therefore, it is only natural to ask if some POMDP solver (Pineau, Gordon, and Thrun 2003) could solve the problems in our experimental section as efficiently as `LinOASM`. We believe that this is unlikely. First, the number of states in our problems is huge, 2^{81} and 2^{500}, far out of reach of the state-of-the-art solvers. Second, note that we learn the model of the problem, not just a policy for

solving a known problem. Most of the work in the POMDP community addresses only the latter problem.

The quality of LinOASM policies naturally depends on the quality of features. We note that LinOASM reduces to OASM (Gabillon et al. 2013) when the feature vector of item i is an indicator vector whose i-th entry is 1. Therefore, LinOASM can be always parameterized such that it performs no worse than OASM.

References

Abbasi-Yadkori, Y.; Pál, D.; and Szepesvári, C. 2011. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems 24*, 2312–2320.

Auer, P. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3:397–422.

Bhamidipati, S.; Kveton, B.; and Muthukrishnan, S. 2013. Minimal interaction search: Multi-way search with item categories. In *Proceedings of AAAI Workshop on Intelligent Techniques for Web Personalization and Recommendation*.

Butko, N., and Movellan, J. 2009. Optimal scanning for faster object detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Dani, V.; Hayes, T.; and Kakade, S. 2008. Stochastic linear optimization under bandit feedback. In *Proceedings of the 21st Annual Conference on Learning Theory*, 355–366.

Filippi, S.; Cappé, O.; Garivier, A.; and Szepesvári, C. 2010. Parametric bandits: The generalized linear case. In *Advances in Neural Information Processing Systems 23*, 586–594.

Filippi, S. 2010. *Optimistic Strategies in Reinforcement Learning*. Ph.D. Dissertation, École nationale supérieure des télécommunications - ENST.

Gabillon, V.; Kveton, B.; Wen, Z.; Eriksson, B.; and Muthukrishnan, S. 2013. Adaptive submodular maximization in bandit setting. In *Advances in Neural Information Processing Systems 26*, 2697–2705.

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

Guillory, A., and Bilmes, J. 2011. Simultaneous learning and covering with adversarial noise. In *Proceedings of the 28th International Conference on Machine Learning*, 369–376.

Kempe, D.; Kleinberg, J.; and Tardos, É. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 137–146.

Koren, Y.; Bell, R.; and Volinsky, C. 2009. Matrix factorization techniques for recommender systems. *IEEE Computer* 42(8):30–37.

Krause, A.; Singh, A. P.; and Guestrin, C. 2008. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research* 9:235–284.

Lam, S., and Herlocker, J. 2013. MovieLens 1M Dataset. http://www.grouplens.org/node/12.

Li, L.; Chu, W.; Langford, J.; and Schapire, R. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*.

McCullagh, P., and Nelder, J. A. 1989. *Generalized Linear Models*. Chapman & Hall.

Nemhauser, G. L.; Wolsey, L. A.; and Fisher, M. L. 1978. An analysis of approximations for maximizing submodular set functions - I. *Mathematical Programming* 14(1):265–294.

2013. Netflix. http://www.netflix.com.

Pineau, J.; Gordon, G.; and Thrun, S. 2003. Point-based value iteration: An anytime algorithm for POMDPs. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, 1025–1032.

Sondik, E. 1971. *The Optimal Control of Partially Observable Markov Decision Processes*. Ph.D. Dissertation, Stanford University.

Sutton, R., and Barto, A. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Tarabanis, K.; Allen, P.; and Tsai, R. 1995. A survey of sensor planning in computer vision. *IEEE Transactions on Robotics and Automation* 11(1):86–104.

UCSD. 2013. MPLab GENKI Database. http://mplab.ucsd.edu.

Vijayanarasimhan, S., and Kapoor, A. 2010. Visual recognition and detection under bounded computational resources. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Wen, Z.; Kveton, B.; Eriksson, B.; and Bhamidipati, S. 2013. Sequential Bayesian search. In *Proceedings of the 30th International Conference on Machine Learning*, 977–983.

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