Learning Policies for Contextual Submodular Prediction

Stephane Ross STEPHANEROSS@CMU.EDU
Jiaji Zhou JIAJIZ@ANDREW.CMU.EDU
Yisong Yue YISONGYUE@CMU.EDU
Debadeepta Dey DEBADEEP@CS.CMU.EDU
J. Andrew Bagnell DBAGNELL@RI.CMU.EDU
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

Many prediction domains, such as ad placement, recommendation, trajectory prediction, and document summarization, require predicting a *set* or *list* of options. Such lists are often evaluated using submodular reward functions that measure both quality and diversity. We propose a simple, efficient, and provably near-optimal approach to optimizing such prediction problems based on no-regret learning. Our method leverages a surprising result from online submodular optimization: a single no-regret online learner can compete with an optimal *sequence* of predictions. Compared to previous work, which either learn a sequence of classifiers or rely on stronger assumptions such as realizability, we ensure both data-efficiency as well as performance guarantees in the fully agnostic setting. Experiments validate the efficiency and applicability of the approach on a wide range of problems including manipulator trajectory optimization, news recommendation and document summarization.

1. Introduction

Many problem domains, ranging from web applications such as ad placement or content recommendation to identifying successful robotic grasp trajectories require predicting lists of items. Such applications are often budget-limited and the goal is to choose the best list of items, from a large set of possible items, with maximal utility. In ad placement, we must pick a small set of ads with high click-through rate. For robotic manipulation, we must pick a small set of initial grasp trajectories to maximize the chance of finding a successful trajectory via more extensive evaluation or simulation.

In all of these problems, the predicted list of items should be both relevant and diverse. For example, recommending a diverse set of news articles increases the chance that a user would like at least one article (Radlinski et al., 2008). As such, recommending multiple redundant articles on the same topic would do little to increase this chance. This notion of diminishing returns due to redundancy is often captured formally using submodularity (Guestrin & Krause).

Exact submodular function optimization is intractable, but simple greedy selection is known to have strong near-optimal performance guarantees and typically works very well in practice (Guestrin & Krause). Given access to the submodular reward function, one could simply employ greedy to construct good lists.

In this paper, we study the general supervised learning problem of training a policy to maximize a submodular reward function. We assume that the submodular reward function is only directly measured on a finite training set, and our goal is to learn to make good predictions on new test examples where the reward function is not directly measurable.

We develop a novel agnostic learning approach based on new analysis showing that a **single** no-regret learner can produce a near-optimal *list* of predictions.[1] We use a reduction approach to "lift" this result to contextual hypothesis classes that map features to predictions, and bound performance relative to the optimal sequence of hypotheses in the class. In contrast to previous work, our approach ensures both data-efficiency as well as performance guarantees in the fully

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

[1] This result may seem surprising given that previous approaches (Streeter & Golovin, 2008) require a sequence of online learners – one for each position in the list.

agnostic setting. Moreover, our approach is simple to implement and easily integrates with conventional off-the-shelf learning algorithms. Empirical evaluations show our approach to be competitive with or exceed the state-of-the-art performance on a variety of problems, ranging from trajectory prediction in robotics to extractive document summarization.

2. Related Work

The problem of learning to optimize submodular reward functions from data, both with and without contextual features, has become increasingly important in machine learning due to its diverse application areas. Broadly speaking, there are two main approaches for this setting. The first aims to identify a model within a parametric family of submodular functions and then use the resulting model for new predictions. The second attempts to learn a strategy to directly predict a list of elements by decomposing the overall problem into multiple simpler learning tasks.

The first approach (Yue & Joachims, 2008; Yue & Guestrin, 2011; Lin & Bilmes, 2012; Raman et al., 2012) involves identifying the parameterization that best matches the submodular rewards of the training instances. These methods are largely limited to learning non-negative linear combinations of features that are themselves submodular, which often restricts their expressiveness. Furthermore, while good sample complexity results are known, these guarantees only hold under strong realizability assumptions where submodular rewards can be modeled exactly by such linear combinations (Yue & Guestrin, 2011; Raman et al., 2012). Recent work on *Determinental Point Processes* (DPPs) (Kulesza & Taskar, 2011) provide a probabilistic model of sets, which can be useful for the tasks that we consider. These approaches, while appealing, solve a potentially unnecessarily hard problem in first learning a holistic list evaluation model, and thus may compound errors by first approximating the submodular function and then approximately optimizing it.

The second, a learning reduction approach, by contrast, decomposes list prediction into a sequence of simpler learning tasks that attempts to mimic the greedy strategy (Streeter & Golovin, 2008; Radlinski et al., 2008; Streeter et al., 2009; Dey et al., 2012). In (Dey et al., 2012), this strategy was extended to the contextual setting by a reduction to cost-sensitive classification. Essentially, each learning problem aims to best predict an item to add to the list, given features, so as to maximize the expected marginal utility. This approach is flexible, in that it can be used with most common hypothesis classes and arbitrary features. Because of this decomposition, the full model class (all possible sequences of predictors) is often quite expressive, and allows for agnostic learning guarantees.[2] This generality comes at the expense of being significantly less data-efficient than methods that make realizability assumptions such as (Yue & Guestrin, 2011; Raman et al., 2012), as the existing approach learns a *different* classifier for each position in the list.

Compared with related work, our approach enjoys the benefits of being both data-efficient while ensuring strong agnostic performance guarantees. We do so by developing new analysis for online submodular optimization which yields agnostic learning guarantees while learning a **single** data-efficient policy.

3. Background

Let $\mathcal{S}$ denote the set of possible items to choose from (**e.g.** ads, sentences, grasps). Our objective is to pick a list of items $L \subseteq \mathcal{S}$ to maximize a reward function f that obeys the following properties:[3]

1. **Monotonicity:** For any lists L_1, L_2, $f(L_1) \leq f(L_1 \oplus L_2)$ and $f(L_2) \leq f(L_1 \oplus L_2)$
2. **Submodularity:** For any lists L_1, L_2 and item $s \in \mathcal{S}$, $f(L_1 \oplus s) - f(L_1) \geq f(L_1 \oplus L_2 \oplus s) - f(L_1 \oplus L_2)$.

Here, $\oplus$ denotes the concatenation operator. Intuitively, monotonicity implies that adding more elements never hurts, and submodularity captures the notion of diminishing returns (*i.e.* adding an item to a long list increases the objective less than when adding it to a shorter sublist). We further assume for simplicity that f takes values in $[0, 1]$, and that $f(\emptyset) = 0$ where $\emptyset$ denotes the empty list. We will also use the shorthand $b(s|L) = f(L \oplus s) - f(L)$ to denote the marginal benefit of adding the item s to list L.

A simple example submodular function that repeatedly arises in many domains is one that takes value 0 until a suitable instance is found, and then takes on value 1 thereafter. Examples include the notion of "multiple choice" learning as in (Dey et al., 2012; Guzman-Rivera et al., 2012) where a predicted set of options is considered successful if any predicted item is deemed correct, and abandonment in ad placement (Radlinski et al., 2008) where success is measured by

[2] This first strategy of learning the parameters of a submodular function can be seen as a special case of this second approach (see section 5.1).

[3] "Lists" generalize the notion of "set" more commonly used in submodular optimization, and enables reasoning about item order and repeated items (Streeter & Golovin, 2008). One may consider sets where appropriate.

whether any predicted advertisement is clicked on.

We consider reward functions that may depend on some underlying state $x \in \mathcal{X}$ (e.g. a user, environment of the robot, a document, etc.). Let f_x denote the reward function for state x, and assume that f_x is monotone submodular for all x.

3.1. Learning Problem

Our task consists in learning to construct good lists of pre-specified length k under some unknown distribution of states D (e.g. distribution of users or documents we have to summarize). We consider two cases: context-free and contextual.

Context-Free. In the context-free case, we have no side-information about the current state (i.e. we do not observe anything about x). We quantify the performance of any list L by its expected value:

$$F(L) = \mathbb{E}_{x\sim D}[f_x(L)].$$

Note that $F(L)$ is also monotone submodular. Thus the clairvoyant greedy algorithm with perfect knowledge of D can find a list $\hat{L}_k$ such that $F(\hat{L}_k) \geq (1-1/e)F(L^*_k)$, were $L^*_k = \text{argmax}_{L:|L|=k} F(L)$. Although D is unknown, we assume that we observe samples of the objective f_x during training. Our goal is thus to develop a learning approach that efficiently converges, both computationally and statistically, to the performance of the clairvoyant greedy algorithm.

Contextual. In the contextual case, we observe side-information in the form of features regarding the state of the world. We "lift" this problem to a hypothesis space of policies (*i.e.* multi-class predictors) that map features to items.

Let Π denote our policy class, and let $\pi(x)$ denote the prediction of policy $\pi \in \Pi$ given side-information describing state x. Let $L_{\pi,k} = (\pi_1, \pi_2, \ldots, \pi_k)$ denote a list of policies. In state x, this list of policies will predict $L_{\pi,k}(x) = (\pi_1(x), \pi_2(x), \ldots, \pi_k(x))$. We quantify performance using the expected value:

$$F(L_\pi) = \mathbb{E}_{x\sim D}[f_x(L_\pi(x))].$$

It can be shown that F obeys both monotonicity and submodularity with respect to appending policies (Dey et al., 2012). Thus, a clairvoyant greedy algorithm that sequentially picks the *policy* with highest expected benefit will construct a list $\hat{L}_{\pi,k}$ such that $F(\hat{L}_{\pi,k}) \geq (1-1/e)F(L^*_{\pi,k})$, where $L^*_{\pi,k} = \text{argmax}_{L_\pi:|L_\pi|=k} F(L_\pi)$. As before, our goal is to develop a learning approach (for learning a list of policies) that *efficiently* competes with the performance of the clairvoyant greedy algorithm.

Algorithm 1 Submodular Contextual Policy (SCP) Algorithm in context-free setting.

Input: Set of items $\mathcal{S}$, length m of list to construct, length k of best list to compete against, online learner PREDICT and UPDATE functions.
for $t = 1$ **to** T **do**
  Call online learner PREDICT() m times to construct list L_t. (e.g. by sampling m times from online learner's internal distribution over items).
  Evaluate list L_t on a sampled state $x_t \sim D$.
  For all $s \in \mathcal{S}$, define its discounted cumulative benefit: $r_t(s) = \sum_{i=1}^{m}(1-1/k)^{m-i} b(s|L_{t,i-1}, x_t)$.
  For all $s \in \mathcal{S}$: define loss $\ell_t(s) = \max_{s'\in\mathcal{S}} r_t(s') - r_t(s)$
  Call online learner update with loss ℓ_t: UPDATE(ℓ_t)
end for

4. Context-free List Optimization

We first consider the context-free setting. Our algorithm, called Submodular Contextual Policy (SCP), is described in Algorithm 1. SCP requires an online learning algorithm subroutine (denoted by UPDATE) that is no-regret with respect to a bounded positive loss function,[4] maintains an internal distribution over items for prediction, and can be queried for multiple predictions (i.e. multiple samples).[5] In contrast to prior work (Streeter & Golovin, 2008), SCP employs only a *single* online learning in the inner loop.

SCP proceeds by training over a sequence of states $x_1, x_2, \ldots, x_T$. At each iteration, SCP queries the online learner to generate a list of m items (via PREDICT, e.g. by sampling from its internal distribution over items), evaluates a weighted cumulative benefit of each item on the sampled list to define a loss related to each item, and then uses the online learner (via UPDATE) to update its internal distribution.

During training, we allow the algorithm to construct lists of length m, rather than k. In its simplest form, one may simply choose $m = k$. However, it may be beneficial to choose m differently than k, as is shown later in the theoretical analysis.

Perhaps the most unusual aspect is how loss is defined using the weighted cumulative benefits of each item:

$$r_t(s) = \sum_{i=1}^{m}(1-1/k)^{m-i} b(s|L_{t,i-1}, x_t), \qquad (1)$$

where $L_{t,i-1}$ denotes the first $i-1$ items in L_t, and

$$b(s|L_{t,i-1}, x_t) = f_{x_t}(L_{t,i-1} \oplus s) - f_{x_t}(L_{t,i-1}). \qquad (2)$$

[4] See Section 4.1 and (3) for a definition of no-regret.

[5] Algorithms that meet these requirements include Randomized Weighted Majority (Littlestone & Warmuth, 1994), Follow the Leader (Kalai & Vempala, 2005), EXP3 (Auer et al., 2003), and many others.

Intuitively, (1) represents the weighted sum of benefits of item s in state x_t had we added it at any intermediate stage in L_t. The benefits at different positions are weighed differently, where position i is adjusted by a factor $(1-1/k)^{m-i}$. These weights are derived via our theoretical analysis, and indicate that benefits in early positions should be more discounted than benefits in later positions. Intuitively, this weighting has the effect of rebalancing the benefits so that each position contributes more equally to the overall loss.[6]

SCP requires the ability to directly measure f_x in each training instance x_t. Directly measuring f_{x_t} enables us to obtain loss measurements $\ell_t(s)$ for any $s \in \mathcal{S}$. For example, in document summarization f_x corresponds to the ROUGE score (Lin, 2004), which can be evaluated for any generated summary given expert annotations which are only available for training instances.

In principle, SCP can also be applied in partial feedback settings, e.g. ad placement where the value f_{x_t} is only observed for some items (*e.g.* only the displayed ads), by using bandit learning algorithms instead (e.g. EXP3 (Auer et al., 2003)).[7] As this is an orthogonal issue, most of our focus is on the full information case.

4.1. Theoretical Guarantees

We now show that Algorithm 1 is no-regret with respect to the clairvoyant greedy algorithm's expected performance over the training instances. Our main theoretical result provides a reduction to an online learning problem and directly relates the performance of our algorithm on the submodular list optimization problem to the standard online learning regret incurred by the subroutine.

Although Algorithm 1 uses only a *single* instance of an online learner subroutine, it achieves the same performance guarantee as prior work (Streeter & Golovin, 2008; Dey et al., 2012) that employ k separate instances of an online learner. This leads to a surprising fact: it is possible to sample from a stationary distribution over items to construct a list that achieves the same guarantee as the clairvoyant greedy algorithm. [8]

[6] We also consider a similar algorithm in the min-sum cover setting, where the theory also requires reweighting benefits, but instead weights earlier benefits more highly (by a factor $m-i$, rather than $(1-1/k)^{m-i}$). We omit discussing this variant for brevity.

[7] Partial information settings arise, e.g., when f is derived using real-world trials that preclude the ability to evaluate $b(s|L,x)$ (2) for every possible $s \in \mathcal{S}$.

[8] This fact can also be seen as a special case of a more general result proven in prior related work that analyzed randomized set selection strategies to optimize submodular functions (Feige et al., 2011).

For a sequence of training states $\{x_t\}_{t=1}^T$, let the sequence of loss functions $\{\ell_t\}_{t=1}^T$ defined in Algorithm 1 correspond to the sequence of losses incurred in the reduction to the online learning problem. The expected regret of the online learning algorithm is

$$\mathbb{E}[R] = \sum_{t=1}^{T} \mathbb{E}_{s' \sim p_t}[\ell_t(s')] - \min_{s \in \mathcal{S}} \sum_{t=1}^{T} \ell_t(s), \tag{3}$$

where p_t is the internal distribution of the online learner used to construct list L_t. Note that an online learner is called *no-regret* if R is sublinear in T.

Let $F(p,m) = \mathbb{E}_{L_m \sim p}[\mathbb{E}_{x \sim D}[f_x(L_m)]]$ denote the expected value of constructing lists by sampling (with replacement) m elements from distribution p, and let $\hat{p} = \arg\max_{t \in \{1,2,\ldots,T\}} F(p_t, m)$ denote the best distribution found by the algorithm.

We define a mixture distribution $\overline{p}$ over lists that constructs a list as follows: sample an index t uniformly in $\{1,2,\ldots,T\}$, then sample m elements (with replacement) from p_t. Note that $F(\overline{p},m) = \frac{1}{T}\sum_{t=1}^{T} F(p_t,m)$ and $F(\hat{p},m) \geq F(\overline{p},m)$. Thus it suffices to show that $F(\overline{p},m)$ has good guarantees. We show that in expectation $\overline{p}$ (and thus $\hat{p}$) constructs lists with performance guarantees close to the clairvoyant greedy algorithm:[9]

Theorem 1. *Let* $\alpha = \exp(-m/k)$ *and* $k' = \min(m,k)$. *For any* $\delta \in (0,1)$, *with probability* $\geq 1-\delta$:

$$F(\overline{p},m) \geq (1-\alpha)F(L_k^*) - \frac{\mathbb{E}[R]}{T} - 3\sqrt{\frac{2k'\ln(2/\delta)}{T}}$$

Corollary 1. *If a no-regret algorithm is used on the sequence of loss* ℓ_t, *then as* $T \to \infty$, $\frac{\mathbb{E}[R]}{T} \to 0$, *and:*

$$\lim_{T\to\infty} F(\overline{p},m) \geq (1-\alpha)F(L_k^*)$$

Theorem 1 provides a general approximation ratio to the best list of size k when constructing a list of a different size m. For $m=k$, we obtain the typical $(1-1/e)$ approximation ratio (Guestrin & Krause). As m increases, this provides approximation ratios that converge exponentially closer to 1.

Naively, one might expect regret $\mathbb{E}[R]/T$ to scale linearly in k' as it involves loss in $[0,k']$. However, we show that regret actually scales as $O(\sqrt{k'})$ (e.g. using Weighted Majority (Kalai & Vempala, 2005)). Our result matches the best known results for this setting

[9] Additionally, if the distributions p_t converge, then the last distribution p_{T+1} must have performance arbitrarily close to $\overline{p}$ as $T \to \infty$. In particular, we can expect this to occur when the examples are randomly drawn from a fixed distribution that does not change over time.

Algorithm 2 Submodular Contextual Policy (SCP) Algorithm.

Input: Set of items $\mathcal{S}$, policy class $\tilde{\Pi}$, length m of list we construct, length k of best list we compete against.
Pick initial policy π_1 (or distribution over policies)
for $t = 1$ **to** T **do**
  Observe features of a sampled state $x_t \sim D$ (e.g. features of user/document)
  Construct list L_t of m items using π_t with features of x_t (or by sampling a policy for each position if π_t is a distribution over policies).
  Define m new cost-sensitive classification examples $\{(v_{ti}, c_{ti}, w_{ti})\}_{i=1}^m$ where:

1. v_{ti} is the feature vector of state x_t and list $L_{t,i-1}$
2. c_{ti} is the cost vector such that $\forall s \in \mathcal{S}$: $c_{ti}(s) = \max_{s' \in \mathcal{S}} b(s'|L_{t,i-1}, x_t) - b(s|L_{t,i-1}, x_t)$
3. $w_{ti} = (1 - 1/k)^{m-i}$ is the weight of this example

  $\pi_{t+1} = \textsc{Update}(\pi_t, \{(v_{ti}, c_{ti}, w_{ti})\}_{i=1}^m)$
end for
return π_{T+1}

(Streeter & Golovin, 2008) while using a *single* online learner, and is especially beneficial in the contextual setting due to improved generalization (see Section 5).

Corollary 2. *Using weighted majority with the optimal learning rate guarantees with probability* $\geq 1 - \delta$*:*

$$F(\overline{p}, m) \geq (1-\alpha)F(L_k^*) - O\left(\sqrt{\frac{k' \log(1/\delta)}{T}} + \sqrt{\frac{k' \log |\mathcal{S}|}{T}}\right).$$

5. Contextual List Optimization with Stationary Policies

We now consider the contextual setting where features of each state x_t are observed before choosing the list. As mentioned, our goal here is to compete with the best list of policies $(\pi_1, \pi_2, \ldots, \pi_k)$ from a hypothesis class Π. Each of these policies are assumed to choose an item solely based on features of the state x_t.

We consider embedding Π within a larger class, $\Pi \subseteq \tilde{\Pi}$, where policies $\tilde{\Pi}$ are functions of both state and a partially chosen list. Then for any $\pi \in \tilde{\Pi}$, $\pi(x, L)$ corresponds to the item that policy π selects to append to list L given state x. We will learn a policy, or distribution of policies, from $\tilde{\Pi}$ that attempts to generalize list construction across multiple positions.[10]

We present an extension of SCP to the contextual setting (Algorithm 2). At each iteration, SCP constructs a list L_t for the state x_t (using its current policy or by sampling policies from its distribution over policies).

Analogous to the context-free setting, we define a loss function for the learner subroutine (Update). We represent the loss using weighted cost-sensitive classification examples $\{(v_{ti}, c_{ti}, w_{ti})\}_{i=1}^m$, where v_{ti} denotes features of the state x_t and list $L_{t,i-1}$, $w_{ti} = (1-1/k)^{m-i}$ is the weight associated to this example, and c_{ti} is the cost vector specifying the cost of each item $s \in \mathcal{S}$

$$c_{ti}(s) = \max_{s' \in \mathcal{S}} b(s'|L_{t,i-1}, x_t) - b(s|L_{t,i-1}, x_t). \quad (4)$$

The loss incurred by any policy π is defined by its loss on this set of cost-sensitive classification examples, i.e.

$$\ell_t(\pi) = \sum_{i=1}^m w_{ti} c_{ti}(\pi(v_{ti})).$$

These new examples are then used to update the policy (or distribution over policies) using a no-regret algorithm (Update). This reduction effectively transforms the task of learning a policy for this submodular list optimization problem into a standard online cost-sensitive classification problem.[11] Analogous to the context-free setting, we can also extend to partial feedback settings where f is only partially measurable by using contextual bandit algorithms such as EXP4 (Auer et al., 2003) as the online learner (Update).[12]

5.1. No-Regret Cost-Sensitive Classification

Having transformed our problem into online cost-sensitive classification, we now present approaches that can be used to achieve no-regret on such tasks. For finite policy classes $\tilde{\Pi}$, one can again leverage any no-regret online algorithm such as Weighted Majority (Kalai & Vempala, 2005). Weighted Majority maintains a distribution over policies in $\tilde{\Pi}$ based on the loss $\ell_t(\pi)$ of each π, and achieves regret at a rate of

$$R = \sqrt{k' \log |\tilde{\Pi}| / T},$$

for $k' = \min(m, k)$. In fact, the context-free setting can be seen as a special case, where $\Pi = \tilde{\Pi} = \{\pi_s | s \in \mathcal{S}\}$ and $\pi_s(v) = s$ for any v.

[10] Competing against the best list of policies in $\tilde{\Pi}$ is difficult in general as it violates submodularity: policies can perform better when added later in the list (due to list features). Nevertheless, we can still learn from class $\tilde{\Pi}$ and compete against the best list of policies in Π.

[11] This is similar to DAgger (Ross et al., 2011a;b; Ross & Bagnell, 2012) developed for sequential prediction problems like imitation learning. Our work can be seen as a specialization of DAgger for submodular list optimization, and ensures that we learn policies that pick good items under the lists they construct. Unlike prior work, our analysis leverages submodularity, leading to several modifications, and improved global optimality guarantees.

[12] Analogous to the context-free setting, partial information arises when c_{ti} (4) is not measurable for every $s \in \mathcal{S}$.

However, achieving no-regret for infinite policy classes is in general not tractable. A more practical approach is to employ existing reductions of cost-sensitive classification problems to convex optimization problems, for which we can efficiently run no-regret convex optimization (e.g. gradient descent). These reductions effectively upper bound the cost-sensitive loss by a convex loss, and thus bound the original loss of the list prediction problem. We briefly describe two such reductions from (Beygelzimer et al., 2005):

Reduction to Regression We transform cost-sensitive classification into a regression problem of predicting the costs of each item $s \in \mathcal{S}$. Afterwards, the policy chooses the item with lowest predicted cost. We convert each weighted cost-sensitive example (v, c, w) into $|\mathcal{S}|$ weighted regression examples.

For example, if we use least-squares linear regression, the weighted squared loss for a particular example (v, c, w) and policy h would be:

$$\ell(h) = w \sum_{s \in \mathcal{S}} (h^\top v(s) - c(s))^2.$$

Reduction to Ranking Another useful reduction transforms the problem into a "ranking" problem that penalizes ranking an item s above another better item s'. In our experiments, we employ a weighted hinge loss, and so the penalty is proportional to the difference in cost of the misranked pair. For each cost-sensitive example (v, c, w), we generate $|\mathcal{S}|(|\mathcal{S}| - 1)/2$ ranking examples for every distinct pair of items (s, s'), where we must predict the best item among (s, s') (potentially by a margin), with a weight of $w|c(s) - c(s')|$.

For example, if we train a linear SVM (Joachims, 2005), we obtain a weighted hinge loss of the form:

$$w|\delta_{s,s'}| \max(0, 1 - h^\top (v(s) - v(s'))\text{sign}(\delta_{s,s'})),$$

where $\delta_{s,s'} = c(s) - c(s')$ and h is the linear policy. At prediction time, we simply predict the item s^* with highest score, $s^* = \text{argmax}_{s \in \mathcal{S}} h^\top v(s)$. This reduction proves advantageous whenever it is easier to predict pairwise rankings rather than the actual cost.

5.2. Theoretical Guarantees

We now present contextual performance guarantees for SCP that relate performance on the submodular list optimization task to the regret of the corresponding online cost-sensitive classification task. Let $\ell_t : \tilde{\Pi} \to \mathbb{R}$ compute the loss of each policy π on the cost-sensitive classification examples $\{v_{ti}, c_{ti}, w_{ti}\}_{i=1}^m$ collected in Algorithm 2 for state x_t. We use $\{\ell_t\}_{t=1}^T$ as the sequence of losses for the online learning problem.

For a deterministic online algorithm that picks the sequence of policies $\{\pi_t\}_{t=1}^T$, the regret is

$$R = \sum_{t=1}^T \ell_t(\pi_t) - \min_{\pi \in \tilde{\Pi}} \sum_{t=1}^T \ell_t(\pi).$$

For a randomized online learner, let π_t be the distribution over policies at iteration t, with expected regret

$$\mathbb{E}[R] = \sum_{t=1}^T \mathbb{E}_{\pi'_t \sim \pi_t}[\ell_t(\pi'_t)] - \min_{\pi \in \tilde{\Pi}} \sum_{t=1}^T \ell_t(\pi).$$

Let $F(\pi, m) = \mathbb{E}_{L_{\pi,m} \sim \pi}[\mathbb{E}_{x \sim D}[f_x(L_{\pi,m}(x))]]$ denote the expected value of constructing lists by sampling (with replacement) m policies from distribution π (if π is a deterministic policy, then this means we use the same policy at each position in the list). Let $\hat{\pi} = \text{argmax}_{t \in \{1,2,\ldots,T\}} F(\pi_t, m)$ denote the best distribution found by the algorithm in hindsight.

We use a mixture distribution $\overline{\pi}$ over policies to construct a list as follows: sample an index t uniformly in $\{1, 2, \ldots, T\}$, then sample m policies from π_t to construct the list. As before, we note that $F(\overline{\pi}, m) = \frac{1}{T} \sum_{t=1}^T F(\pi_t, m)$, and $F(\hat{\pi}, m) \geq F(\overline{\pi}, m)$. As such, we again focus on proving good guarantees for $F(\overline{\pi}, m)$, as shown by the following theorem.

Theorem 2. *Let $\alpha = \exp(-m/k)$, $k' = \min(m, k)$ and pick any $\delta \in (0, 1)$. After T iterations, for deterministic online algorithms, we have that with probability at least $1 - \delta$:*

$$F(\overline{\pi}, m) \geq (1 - \alpha) F(L^*_{\pi,k}) - \frac{R}{T} - 2\sqrt{\frac{2 \ln(1/\delta)}{T}}.$$

Similarly, for randomized online algorithms, with probability at least $1 - \delta$:

$$F(\overline{\pi}, m) \geq (1 - \alpha) F(L^*_{\pi,k}) - \frac{\mathbb{E}[R]}{T} - 3\sqrt{\frac{2k' \ln(2/\delta)}{T}}.$$

Thus, as in the previous section, a no-regret algorithm must achieve $F(\overline{\pi}, m) \geq (1 - \alpha) F(L^*_{\pi,k})$ with high probability as $T \to \infty$. This matches similar guarantees provided in (Dey et al., 2012). Despite having similar guarantees, we intuitively expect SCP to outperform (Dey et al., 2012) in practice because SCP can use all data to train a *single* predictor, instead of being split to train k separate ones. We empirically verify this intuition in Section 6.

When using surrogate convex loss functions (such as regression or ranking loss), we provide a general result that applies if the online learner uses any convex upper bound of the cost-sensitive loss. An extra penalty term is introduced that relates the gap between the convex upper bound and the original cost-sensitive loss:

Corollary 3. *Let* $\alpha = \exp(-m/k)$ *and* $k' = \min(m, k)$. *If we run an online learning algorithm on the sequence of convex loss* C_t *instead of* ℓ_t, *then after* T *iterations, for any* $\delta \in (0, 1)$, *we have that with probability at least* $1 - \delta$:

$$F(\overline{\pi}, m) \geq (1 - \alpha)F(L^*_{\pi,k}) - \frac{\tilde{R}}{T} - 2\sqrt{\frac{2\ln(1/\delta)}{T}} - \mathcal{G}$$

where $\tilde{R}$ *is the regret on the sequence of convex loss* C_t, *and* $\mathcal{G}$ *is defined as*

$$\frac{1}{T}\left[\sum_{t=1}^{T}(\ell_t(\pi_t) - C_t(\pi_t)) + \min_{\pi\in\tilde{\Pi}}\sum_{t=1}^{T} C_t(\pi) - \min_{\pi'\in\tilde{\Pi}}\sum_{t=1}^{T}\ell_t(\pi')\right]$$

and denotes the "convex optimization gap" that measures how close the surrogate C_t *is to minimizing* ℓ_t.

This result implies that using a good surrogate convex loss for no-regret convex optimization will lead to a policy that has a good performance relative to the optimal list of policies. Note that the gap $\mathcal{G}$ often may be small or non-existent. For instance, in the case of the reduction to regression or ranking, $\mathcal{G} = 0$ in realizable settings where there exists a "perfect" predictor in the class. Similarly, in cases where the problem is near-realizable we would expect $\mathcal{G}$ to be small.[13]

6. Experimental Results

6.1. Robotic Manipulation Planning

We applied SCP to a manipulation planning task for a 7 degree-of-freedom robot manipulator. The goal is to predict a set of initial trajectories so as to maximize the chance that one of them leads to a collision-free trajectory. We use local trajectory optimization techniques such as CHOMP (Ratliff et al., 2009), which have proven effective in quickly finding collision-free trajectories using local perturbations of an initial trajectory. Note that selecting a diverse set of initial trajectories is important since local techniques such as CHOMP often get stuck in local optima.[14]

We use the dataset from (Dey et al., 2012). It consists of 310 training and 212 test environments of random obstacle configurations around a target object, and 30 initial seed trajectories. In each environment, each seed trajectory has 17 features describing the spatial properties of the trajectory relative to obstacles.[15]

Following (Dey et al., 2012), we employ a reduction of cost-sensitive classification to regression as explained in Section 5.1. We compare SCP to ConSeqOpt (Dey et al., 2012) (which learns k separate predictors), and Regression (regress success rate from features to sort seeds; this accounts for relevance but not diversity).

Figure 1 (left) shows the failure probability over the test environments versus the number of training environments. ConSeqOpt employs a reduction to k classifiers. As a consequence, ConSeqOpt faces data starvation issues for small training sizes, as there is little data available for training predictors lower in the list.[16] In contrast, SCP has no data starvation issue and outperforms both ConSeqOpt and Regression.

6.2. Personalized News Recommendation

We built a stochastic user simulation based on 75 user preferences derived from a user study in (Yue & Guestrin, 2011). Using this simulation as a training oracle, our goal is to learn to recommend articles to any user (depending on their contextual features) to minimize the failure case where the user does not like any of the recommendations.[17]

Articles are represented by features, and user preferences by linear weights. We derived user contexts by soft-clustering users into groups, and using corrupted group memberships as contexts.

We perform five-fold cross validation. In each fold, we train SCP and ConSeqOpt on 40 users' preferences, use 20 users for validation, and then test on the held-out 15 users. Training, validation and testing are all performed via simulation. Figure 1 (middle) shows the results, where we see the recommendations made by SCP achieves significantly lower failure rate as the number of recommendations is increased from 1 to 5.

6.3. Document Summarization

In the extractive multi-document summarization task, the goal is to extract sentences (with character budget B) to maximize coverage of human-annotated summaries. Following the experimental setup from (Lin & Bilmes, 2010) and (Kulesza & Taskar, 2011), we use

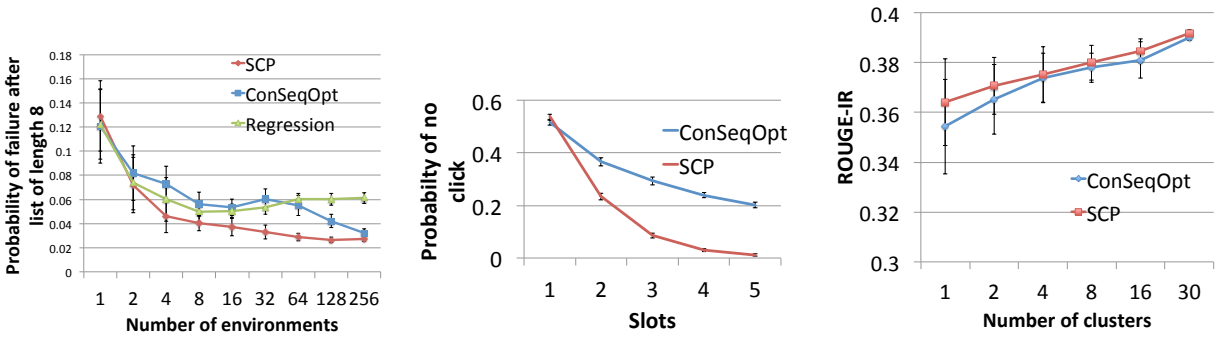
[13] We conjecture that this gap term $\mathcal{G}$ is not specific to our particular scenario, but rather is (implicitly) always present whenever one attempts to optimize classification accuracy via surrogate convex optimization.

[14] I.e., similar or redundant inital trajectories will lead to the same local optima.

[15] In addition to the base features, we add features of the current list w.r.t. each initial trajectory. We use the per feature minimum absolute distance and average absolute value of the distance to the features of initial trajectories in the list. We also use a bias feature always set to 1, and an indicator feature which is 1 when selecting the element in the first position, 0 otherwise.

[16] When a successful seed is found, benefits at later positions are 0. This effectively discards training environments for training classifiers lower in the list in ConSeqOpt.

[17] Also known as abandonment (Radlinski et al., 2008).



(a) (b) (c)

Figure 1. (a)SCP performs better at even low data availability while ConSeqOpt suffers from data starvation issues (b) With increase in slots SCP predicts news articles which have lower probability of the user not clicking on any of them compared to ConSeqOpt (c) ROUGE-1R scores with respect to the size of the training data

data from the Document Understanding Conference (DUC) 2003 and 2004 (Task 2) (Dang, 2005). Each training or test instance corresponds to a cluster of documents, and contains approximately 10 documents belonging to the same topic and four human reference summaries. We train on the 2003 data (30 clusters) and test on the 2004 data (50 clusters). The budget is $B = 665$ bytes, including spaces.

We use the ROUGE (Lin, 2004) unigram statistics (ROUGE-1R, ROUGE-1P, ROUGE-1F) for performance evaluation. Our method directly attempts to optimize the ROUGE-1R objective with respect to the reference summaries, which can be easily shown to be monotone submodular (Lin & Bilmes, 2011).

We aim to predict sentences that are both short and informative. Therefore we maximize the normalized marginal benefit,

$$b'(s|L_{t,i-1}) = b(s|L_{t,i-1})/l(s), \tag{5}$$

where $l(s)$ is the length of the sentence s.[18] We use a reduction to ranking as described in Section 5.1 using (5). While not performance-optimized, our approach takes less than 15 minutes to train.

Following (Kulesza & Taskar, 2011), we consider features f_i for each sentence consisting of *quality features* q_i and *similarity features* ϕ_i ($f_i = [q_i^T, \phi_i^T]^T$). The quality features, attempt to capture the representativeness for a single sentence. Similarity features q_i for sentence s_i as we construct the list L_t measure a notion of distance of a proposed sentence to sentences already included in the set.[19]

[18] This results in a knapsack constrained optimization problem. We expect our approach to perform well in this setting, but defer a formal analysis for future work.

[19] A variety of similarity features were considered, with the simplest being average squared distance of tf-idf vectors. Performance was very stable across different features. The experiments presented use three types: 1) following the idea in (Kulesza & Taskar, 2011) of similarity as a volume metric, we compute the squared volume of the parallelopiped spanned by the TF-IDF vectors of sentences in the set $L_{t,k} \cup s_i$; 2) the product between $\det(G_{L_{t,k} \cup s_i})$ and the quality features; 3) the minimum absolute distance of quality features between s_i and each element in $L_{t,k}$.

System	ROUGE-1F	ROUGE-1P	ROUGE-1R
SubMod	37.39	36.86	37.99
DPP	38.27	37.87	38.71
ConSeqOpt	39.02 ± 0.07	39.08±0.07	39.00±0.12
SCP	**39.15**±0.15	**39.16**±0.15	**39.17**±0.15
Greedy (Oracle)	44.92	45.14	45.24

Table 1. ROUGE unigram score on the DUC 2004 test set

Table 1 shows the performance (Rouge unigram statistics) comparing SCP with existing algorithms. We observe that SCP outperforms existing state-of-the-art approaches, which we denote SubMod (Lin & Bilmes, 2010) and DPP (Kulesza & Taskar, 2011). "Greedy (Oracle)" corresponds to the clairvoyant oracle that directly optimizes the test Rouge score and thus serves as an upper bound on this class of techniques. Figure 1 (right) plots Rouge-1R performance as a function of the size of training data, suggesting SCP's superior data-efficiency compared to ConSeqOpt.

Acknowledgements

This research was supported in part by NSF NRI *Purposeful Prediction* project and ONR MURIs *Decentralized Reasoning in Reduced Information Spaces* and *Provably Stable Vision-Based Control*. Yisong Yue was also supported in part by ONR (PECASE) N000141010672 and ONR Young Investigator Program N00014-08-1-0752. We gratefully thank Martial Hebert for valuable discussions and support.

References

Auer, Peter, Cesa-Bianchi, Nicoló, Freund, Yoav, and Schapire, Robert. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32 (1):48–77, 2003.

Beygelzimer, Alina, Dani, Varsha, Hayes, Thomas, Langford, John, and Zadrozny, Bianca. Error limiting reductions between classification tasks. In *ICML*. ACM, 2005.

Dang, Hoa Trang. Overview of duc 2005. In *DUC*, 2005.

Dey, Debadeepta, Liu, Tian Yu, Hebert, Martial, and Bagnell, J. Andrew (Drew). Contextual sequence optimization with application to control library optimization. In *RSS*, 2012.

Feige, U., Mirrokni, V. S., and Vondrak, J. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

Guestrin, Carlos and Krause, Andreas. Beyond convexity: Submodularity in machine learning. URL www.submodularity.org.

Guzman-Rivera, Abner, Batra, Dhruv, and Kohli, Pushmeet. Multiple choice learning: Learning to produce multiple structured outputs. In *NIPS*, 2012.

Joachims, Thorsten. A support vector method for multivariate performance measures. In *ICML*. ACM, 2005.

Kalai, Adam and Vempala, Santosh. Efficient algorithms for online decision problems. *JCSS*, 71(3): 291–307, 2005.

Kulesza, Alex and Taskar, Ben. Learning determinantal point processes. In *UAI*, 2011.

Lin, Chin-Yew. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: ACL-04 Workshop*, 2004.

Lin, Hui and Bilmes, Jeff. Multi-document summarization via budgeted maximization of submodular functions. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2010.

Lin, Hui and Bilmes, Jeff. A class of submodular functions for document summarization. In *ACL-HLT*, 2011.

Lin, Hui and Bilmes, Jeff. Learning mixtures of submodular shells with application to document summarization. In *UAI*, 2012.

Littlestone, Nick and Warmuth, Manfred. The Weighted Majority Algorithm. *INFORMATION AND COMPUTATION*, 1994.

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

Raman, Karthik, Shivaswamy, Pannaga, and Joachims, Thorsten. Online learning to diversify from implicit feedback. In *KDD*, 2012.

Ratliff, Nathan, Zucker, Matt, Bagnell, J. Andrew, and Srinivasa, Siddhartha. Chomp: Gradient optimization techniques for efficient motion planning. In *ICRA*, May 2009.

Ross, Stephane and Bagnell, J. Andrew. Agnostic system identification for model-based reinforcement learning. In *ICML*, 2012.

Ross, Stephane, Gordon, Geoff, and Bagnell, J. Andrew. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 2011a.

Ross, Stephane, Munoz, Daniel, Bagnell, J. Andrew, and Hebert, Martial. Learning message-passing inference machines for structured prediction. In *CVPR*, 2011b.

Streeter, M. and Golovin, D. An online algorithm for maximizing submodular functions. In *NIPS*, 2008.

Streeter, Matthew, Golovin, Daniel, and Krause, Andreas. Online learning of assignments. In *NIPS*, 2009.

Yue, Yisong and Guestrin, Carlos. Linear submodular bandits and their application to diversified retrieval. In *NIPS*, 2011.

Yue, Yisong and Joachims, Thorsten. Predicting diverse subsets using structural svms. In *ICML*, 2008.

Learning Policies for Contextual Submodular Prediction - Supplementary Material

Stephane Ross STEPHANEROSS@CMU.EDU
Jiaji Zhou JIAJIZ@ANDREW.CMU.EDU
Yisong Yue YISONGYUE@CMU.EDU
Debadeepta Dey DEBADEEP@CS.CMU.EDU
J. Andrew Bagnell DBAGNELL@RI.CMU.EDU
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA

A. Proofs of Theoretical Results

This appendix contains the proofs of the various theoretical results presented in this paper.

A.1. Preliminaries

We begin by proving a number of lemmas about monotone submodular functions, which will be useful to prove our main results.

Lemma 1. *Let $\mathcal{S}$ be a set and f be a monotone submodular function defined on list of items from $\mathcal{S}$. For any lists A, B, we have that:*

$$f(A \oplus B) - f(A) \leq |B|(\mathbb{E}_{s\sim U(B)}[f(A \oplus s)] - f(A))$$

for $U(B)$ the uniform distribution on items in B.

Proof. For any list A and B, let B_i denote the list of the first i items in B, and b_i the i^{th} item in B. We have that:

$$\begin{aligned} & f(A \oplus B) - f(A) \\ = & \sum_{i=1}^{|B|} f(A \oplus B_i) - f(A \oplus B_{i-1}) \\ \leq & \sum_{i=1}^{|B|} f(A \oplus b_i) - f(A) \\ = & |B|(\mathbb{E}_{b\sim U(B)}[f(A \oplus b)] - f(A)) \end{aligned}$$

where the inequality follows from the submodularity property of f. □

Lemma 2. *Let $\mathcal{S}$ be a set, and f a monotone submodular function defined on lists of items in $\mathcal{S}$. Let A, B be any lists of items from $\mathcal{S}$. Denote A_j the list of the first j items in A, $U(B)$ the uniform distribution on items in B and define $\epsilon_j = \mathbb{E}_{s\sim U(B)}[f(A_{j-1} \oplus s)] - f(A_j)$, the additive error term in competing with the average marginal benefits of the items in B when picking the j^{th} item in A (which could be positive or negative). Then:*

$$f(A) \geq (1-(1-1/|B|)^{|A|})f(B) - \sum_{i=1}^{|A|}(1-1/|B|)^{|A|-i}\epsilon_i$$

In particular if $|A| = |B| = k$, then:

$$f(A) \geq (1-1/e)f(B) - \sum_{i=1}^{k}(1-1/k)^{k-i}\epsilon_i$$

and for $\alpha = \exp(-|A|/|B|)$ (i.e. $|A| = |B|\log(1/\alpha)$):

$$f(A) \geq (1-\alpha)f(B) - \sum_{i=1}^{|A|}(1-1/|B|)^{|A|-i}\epsilon_i$$

Proof. Using the monotone property and previous lemma 1, we must have that: $f(B) - f(A) \leq f(A \oplus B) - f(A) \leq |B|(\mathbb{E}_{b\sim U(B)}[f(A \oplus b)] - f(A))$.

Now let $\Delta_j = f(B) - f(A_j)$. By the above we have that

$$\begin{aligned} & \Delta_j \\ \leq & |B|[\mathbb{E}_{s\sim U(B)}[f(A_j \oplus s)] - f(A_j)] \\ = & |B|[\mathbb{E}_{s\sim U(B)}[f(A_j \oplus s)] - f(A_{j+1}) \\ & + f(A_{j+1}) - f(B) + f(B) - f(A_j)] \\ = & |B|[\epsilon_{j+1} + \Delta_j - \Delta_{j+1}] \end{aligned}$$

Rearranging terms, this implies that $\Delta_{j+1} \leq (1 - 1/|B|)\Delta_j + \epsilon_{j+1}$. Recursively expanding this recurrence from $\Delta_{|A|}$, we obtain:

$$\Delta_{|A|} \leq (1-1/|B|)^{|A|}\Delta_0 + \sum_{i=1}^{|A|}(1-1/|B|)^{|A|-i}\epsilon_i$$

Using the definition of $\Delta_{|A|}$ and rearranging terms, we obtain $f(A) \geq (1-(1-1/|B|)^{|A|})f(B) - \sum_{i=1}^{|A|}(1-$

$1/|B|)^{|A|-i}\epsilon_i$. This proves the first statement of the theorem. The following two statements follow from the observations that $(1-1/|B|)^{|A|} = \exp(|A|\log(1-1/|B|)) \leq \exp(-|A|/|B|) = \alpha$. Hence $(1-(1-1/|B|)^{|A|})f(B) \geq (1-\alpha)f(B)$. When $|A| = |B|$, $\alpha = 1/e$ and this proves the special case where $|A| = |B|$. □

For the greedy list construction strategy, the ϵ_j in the last lemma are always ≤ 0, such that Lemma 2 implies that if we construct a list of size k with greedy, it must achieve at least 63% of the value of the optimal list of size k, but also that it must achieve at least 95% of the value of the optimal list of size $\lfloor k/3 \rfloor$, and at least 99.9% of the value of the optimal list of size $\lfloor k/7 \rfloor$.

A more surprising fact that follows from the last lemma is that constructing a list stochastically, by sampling items from a particular fixed distribution, can provide the same guarantee as greedy:

Lemma 3. *Let $\mathcal{S}$ be a set, and f a monotone submodular function defined on lists of items in $\mathcal{S}$. Let B be any list of items from $\mathcal{S}$ and $U(B)$ the uniform distribution on elements in B. Suppose we construct the list A by sampling k items randomly from $U(B)$ (with replacement). Denote A_j the list obtained after j samples, and P_j the distribution over lists obtained after j samples. Then:*

$$\mathbb{E}_{A\sim P_k}[f(A)] \geq (1-(1-1/|B|)^k)f(B)$$

In particular, for $\alpha = \exp(-k/|B|)$:

$$\mathbb{E}_{A\sim P_k}[f(A)] \geq (1-\alpha)f(B)$$

Proof. The proof follows a similar proof to the previous lemma. Recall that by the monotone property and lemma 1, we have that for any list A: $f(B) - f(A) \leq f(A \oplus B) - f(A) \leq |B|(\mathbb{E}_{b\sim U(B)}[f(A \oplus b)] - f(A))$. Because this holds for all lists, we must also have that for any distribution P over lists A, $f(B) - \mathbb{E}_{A\sim P}[f(A)] \leq |B|\mathbb{E}_{A\sim P}[\mathbb{E}_{b\sim U(B)}[f(A \oplus b)] - f(A)]$. Also note that by the way we construct sets, we have that $\mathbb{E}_{A_{j+1}\sim P_{j+1}}[f(A_{j+1})] = \mathbb{E}_{A_j\sim P_j}[\mathbb{E}_{s\sim U(B)}[f(A_j \oplus s)]]$

Now let $\Delta_j = f(B) - \mathbb{E}_{A_j\sim P_j}[f(A_j)]$. By the above we have that:

$$\begin{aligned}
&\Delta_j \\
\leq\ & |B|\mathbb{E}_{A_j\sim P_j}[\mathbb{E}_{s\sim U(B)}[f(A_j \oplus s)] - f(A_j)] \\
=\ & |B|\mathbb{E}_{A_j\sim P_j}[\mathbb{E}_{s\sim U(B)}[f(A_j \oplus s)] - f(B) \\
& + f(B) - f(A_j)] \\
=\ & |B|(\mathbb{E}_{A_{j+1}\sim P_{j+1}}[f(A_{j+1})] - f(B) \\
& + f(B) - \mathbb{E}_{A_j\sim P_j}[f(A_j)]) \\
=\ & |B|[\Delta_j - \Delta_{j+1}]
\end{aligned}$$

Rearranging terms, this implies that $\Delta_{j+1} \leq (1-1/|B|)\Delta_j$. Recursively expanding this recurrence from Δ_k, we obtain:

$$\Delta_k \leq (1-1/|B|)^k\Delta_0$$

Using the definition of Δ_k and rearranging terms we obtain $E_{A\sim P_k}[f(A)] \geq (1-(1-1/|B|)^k)f(B)$. The second statement follows again from the fact that $(1-(1-1/|B|)^k)f(B) \geq (1-\alpha)f(B)$ □

Corollary 1. *There exists a distribution that when sampled k times to construct a list, achieves an approximation ratio of $(1-1/e)$ of the optimal list of size k in expectation. In particular, if A^* is an optimal list of size k, sampling k times from $U(A^*)$ achieves this approximation ratio. Additionally, for any $\alpha \in (0,1]$, sampling $\lceil k\log(1/\alpha)\rceil$ times must construct a list that achieves an approximation ratio of $(1-\alpha)$ in expectation.*

Proof. Follows from the last lemma using $B = A^*$. □

This surprising result can also be seen as a special case of a more general result proven in prior related work that analyzed randomized set selection strategies to optimize submodular functions (lemma 2.2 in (Feige et al., 2011)).

A.2. Proofs of Main Results

We now provide the proofs of the main results in this paper. We provide the proofs for the more general contextual case where we learn over a policy class $\tilde{\Pi}$. All the results for the context-free case can be seen as special cases of these results when $\Pi = \tilde{\Pi} = \{\pi_s | s \in \mathcal{S}\}$ and $\pi_s(x, L) = s$ for any state x and list L.

We refer the reader to the notation defined in section 3 and 5 for the definitions of the various terms used.

Theorem 2 . *Let $\alpha = \exp(-m/k)$ and $k' = \min(m,k)$. After T iterations, for any $\delta, \delta' \in (0,1)$, we have that with probability at least $1-\delta$:*

$$F(\overline{\pi}, m) \geq (1-\alpha)F(L^*_{\pi,k}) - \frac{R}{T} - 2\sqrt{\frac{2\ln(1/\delta)}{T}}$$

and similarly, with probability at least $1-\delta-\delta'$:

$$\begin{aligned}
F(\overline{\pi}, m) \geq\ & (1-\alpha)F(L^*_{\pi,k}) - \tfrac{\mathbb{E}[R]}{T} - \sqrt{\tfrac{2k'\ln(1/\delta')}{T}} \\
& -2\sqrt{\tfrac{2\ln(1/\delta)}{T}}
\end{aligned}$$

Proof.

$$\begin{aligned}
&F(\overline{\pi}, m)\\
=\;& \tfrac{1}{T}\textstyle\sum_{t=1}^{T} F(\pi_t, m)\\
=\;& \tfrac{1}{T}\textstyle\sum_{t=1}^{T} \mathbb{E}_{L_{\pi,m}\sim\pi_t}[\mathbb{E}_{x\sim D}[f_x(L_{\pi,m}(x))]]\\
=\;& (1-\alpha)\mathbb{E}_{x\sim D}[f_x(L^*_{\pi,k}(x))]\\
&-[(1-\alpha)\mathbb{E}_{x\sim D}[f_x(L^*_{\pi,k}(x))]\\
&-\tfrac{1}{T}\textstyle\sum_{t=1}^{T} \mathbb{E}_{L_{\pi,m}\sim\pi_t}[\mathbb{E}_{x\sim D}[f_x(L_{\pi,m}(x))]]]
\end{aligned}$$

Now consider the sampled states $\{x_t\}_{t=1}^T$ and the policies $\pi_{t,i}$ sampled i.i.d. from π_t to construct the lists $\{L_t\}_{t=1}^T$ and denote the random variables $X_t = (1-\alpha)(\mathbb{E}_{x\sim D}[f_x(L^*_{\pi,k}(x))] - f_{x_t}(L^*_{\pi,k}(x_t))) - \mathbb{E}_{L_{\pi,m}\sim\pi_t}[\mathbb{E}_{x\sim D}[f_x(L_{\pi,m}(x))]] - f_{x_t}(L_t)]$. If π_t is deterministic, then simply consider all $\pi_{t,i} = \pi_t$. Because the x_t are i.i.d. from D, and the distribution of policies used to construct L_t only depends on $\{x_\tau\}_{\tau=1}^{t-1}$ and $\{L_\tau\}_{\tau=1}^{t-1}$, then the X_t conditioned on $\{X_\tau\}_{\tau=1}^{t-1}$ have expectation 0, and because $f_x \in [0,1]$ for all state $x \in \mathcal{X}$, X_t can vary in a range $r \subseteq [-2,2]$. Thus the sequence of random variables $Y_t = \sum_{i=1}^{t} X_i$, for t =1 to T, forms a martingale where $|Y_t - Y_{t+1}| \leq 2$. By the Azuma-Hoeffding's inequality, we have that $P(Y_T/T \geq \epsilon) \leq \exp(-\epsilon^2 T/8)$. Hence for any $\delta \in (0,1)$, we have that with probability at least $1-\delta$, $Y_T/T \leq 2\sqrt{\frac{2\ln(1/\delta)}{T}}$. Hence we have that with probability at least $1-\delta$:

$$\begin{aligned}
&F(\overline{\pi}, m)\\
=\;& (1-\alpha)\mathbb{E}_{x\sim D}[f_x(L^*_{\pi,k}(x))]\\
&-[(1-\alpha)\mathbb{E}_{x\sim D}[f_x(L^*_{\pi,k}(x))]\\
&-\tfrac{1}{T}\textstyle\sum_{t=1}^{T} \mathbb{E}_{L_{\pi,m}\sim\pi_t}[\mathbb{E}_{x\sim D}[f_x(L_{\pi,m}(x))]]]\\
=\;& (1-\alpha)\mathbb{E}_{x\sim D}[f_x(L^*_{\pi,k}(x))]\\
&-[(1-\alpha)\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(L^*_{\pi,k}(x_t))\\
&-\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(L_t)] - Y_T/T\\
=\;& (1-\alpha)\mathbb{E}_{x\sim D}[f_x(L^*_{\pi,k}(x))]\\
&-[(1-\alpha)\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(L^*_{\pi,k}(x_t))\\
&-\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(L_t)] - 2\sqrt{\tfrac{2\ln(1/\delta)}{T}}
\end{aligned}$$

Let $w_i = (1-1/k)^{m-i}$. From Lemma 2, we have:

$$\begin{aligned}
&(1-\alpha)\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(L^*_{\pi,k}(x_t)) - \tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(L_t)\\
\leq\;& \tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{i=1}^{m} w_i(\mathbb{E}_{\pi\sim U(L^*_{\pi,k})}[f_{x_t}(L_{t,i-1}\oplus\pi(x_t))]\\
&-f_{x_t}(L_{t,i}))\\
=\;& \mathbb{E}_{\pi\sim U(L^*_{\pi,k})}[\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{i=1}^{m} w_i(f_{x_t}(L_{t,i-1}\oplus\pi(x_t))\\
&-f_{x_t}(L_{t,i}))]\\
\leq\;& \max_{\pi\in\Pi}[\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{i=1}^{m} w_i(f_{x_t}(L_{t,i-1}\oplus\pi(x_t))\\
&-f_{x_t}(L_{t,i}))]\\
\leq\;& \max_{\pi\in\tilde{\Pi}}[\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{i=1}^{m} w_i(f(L_{t,i-1}\oplus\pi(x_t))\\
&-f_{x_t}(L_{t,i}))]\\
=\;& R/T
\end{aligned}$$

Hence combining with the previous result proves the first part of the theorem.

Additionally, for the sampled environments $\{x_t\}_{t=1}^T$ and the policies $\pi_{t,i}$, consider the random variables $Q_{m(t-1)+i} = w_i\mathbb{E}_{\pi\sim\pi_t}[f_{x_t}(L_{t,i-1}\oplus\pi(x_t, L_{t,i-1}))] - w_i f_{x_t}(L_{t,i})$. Because each draw of $\pi_{t,i}$ is i.i.d. from π_t, we have that again the sequence of random variables $Z_j = \sum_{i=1}^{j} Q_i$, for $j = 1$ to Tm forms a martingale and because each Q_i can take values in a range $[-w_j, w_j]$ for $j = 1 + \text{mod}(i-1, m)$, we have $|Z_i - Z_{i-1}| \leq w_j$. Since $\sum_{i=1}^{Tm} |Z_i - Z_{i-1}|^2 \leq T\sum_{i=1}^{m}(1-1/k)^{2(m-i)} \leq T\min(k,m) = Tk'$, by Azuma-Hoeffding's inequality, we must have that $P(Z_{Tm} \geq \epsilon) \leq \exp(-\epsilon^2/2Tk')$. Thus for any $\delta' \in (0,1)$, with probability at least $1-\delta'$, $Z_{Tm} \leq \sqrt{2Tk'\ln(1/\delta)}$. Hence combining with the previous result, it must be the case that with probability at least $1-\delta-\delta'$, both $Y_T/T \leq 2\sqrt{\frac{2\ln(1/\delta)}{T}}$ and $Z_{Tm} \leq \sqrt{2Tk'\ln(1/\delta')}$ holds.

Now note that:

$$\begin{aligned}
&\max_{\pi\in\tilde{\Pi}}[\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{i=1}^{m} w_i(f(L_{t,i-1}\oplus\pi(x_t)) - f_{x_t}(L_{t,i}))]\\
=\;& \max_{\pi\in\tilde{\Pi}}[\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{i=1}^{m} w_i(f_{x_t}(L_{t,i-1}\oplus\pi(x_t))\\
&-\mathbb{E}_{\pi'\sim\pi_t}[f(L_{t,i-1}\oplus\pi'(x_t, L_{t,i-1}))])] + Z_{Tm}/T\\
=\;& \mathbb{E}[R]/T + Z_{Tm}/T
\end{aligned}$$

Using this additional fact, and combining with previous results we must have that with probability at least $1-\delta-\delta'$:

$$\begin{aligned}
&F(\overline{\pi}, m)\\
\geq\;& (1-\alpha)F(L^*_{\pi,k}) - [(1-\alpha)\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(L^*_{\pi,k}(x_t))\\
&-\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(L_t)] - 2\sqrt{\tfrac{2\ln(1/\delta)}{T}}\\
\geq\;& (1-\alpha)F(L^*_{\pi,k}) - \mathbb{E}[R]/T - Z_{Tm}/T - 2\sqrt{\tfrac{2\ln(1/\delta)}{T}}\\
\geq\;& (1-\alpha)F(L^*_{\pi,k}) - \mathbb{E}[R]/T - \sqrt{\tfrac{2k'\ln(1/\delta')}{T}}\\
&-2\sqrt{\tfrac{2\ln(1/\delta)}{T}}
\end{aligned}$$

□

We now show that the expected regret must grow with $\sqrt{k'}$ and not k', hen using Weighted Majority with the optimal learning rate (or with the doubling trick).

Corollary 2 . *Under the event where Theorem 2 holds (the event that occurs w.p. $1-\delta-\delta'$), if $\tilde{\Pi}$ is a finite set of policies, using Weighted Majority with the optimal learning rate guarantees that after T iterations:*

$$\begin{aligned}
\mathbb{E}[R]/T \;\leq\;& \tfrac{4k'\ln|\tilde{\Pi}|}{T} + 2\sqrt{\tfrac{k'\ln|\tilde{\Pi}|}{T}}\\
&+2^{9/4}(k'/T)^{3/4}(\ln(1/\delta'))^{1/4}\sqrt{\ln|\tilde{\Pi}|}
\end{aligned}$$

For large enough T in $\Omega(k'(\ln|\tilde{\Pi}| + \ln(1/\delta')))$, we obtain that:

$$\mathbb{E}[R]/T \leq O(\sqrt{\frac{k' \ln|\tilde{\Pi}|}{T}})$$

Proof. We use a similar argument to Streeter & Golovin Lemma 4 (Streeter & Golovin, 2007) to bound $\mathbb{E}[R]$ in the result of theorem 2 . Consider the sum of the benefits accumulated by the learning algorithm at position i in the list, for $i \in 1, 2, \ldots, m$, i.e. let $y_i = \sum_{t=1}^{T} b(\pi_{t,i}(x_t, L_{t,i-1})|x_t, L_{t,i-1})$, where $\pi_{t,i}$ corresponds to the particular sampled policy by Weighted Majority for choosing the item at position i, when constructing the list L_t for state x_t. Note that $\sum_{i=1}^{m}(1-1/k)^{m-i}y_i \leq \sum_{i=1}^{m} y_i \leq T$ by the fact that the monotone submodular function f_x is bounded in $[0,1]$ for all state x. Now consider the sum of the benefits you could have accumulated at position i, had you chosen the best fixed policy in hindsight to construct all list, keeping the policy fixed as the policy is constructed, i.e. let $z_i = \sum_{t=1}^{T} b(\pi^*(x_t, L_{t,i-1})|x_t, L_{t,i-1})$, for $\pi^* = \arg\max_{\pi \in \tilde{\Pi}} \sum_{i=1}^{m}(1-1/k)^{m-i} \sum_{t=1}^{T} b(\pi^*(x_t, L_{t,i-1})|x_t, L_{t,i-1})$ and let $r_i = z_i - y_i$. Now denote $Z = \sqrt{\sum_{i=1}^{m}(1-1/k)^{m-i}z_i}$. We have $Z^2 = \sum_{i=1}^{m}(1-1/k)^{m-i}z_i = \sum_{i=1}^{m}(1-1/k)^{m-i}(y_i + r_i) \leq T + R$, where R is the sample regret incurred by the learning algorithm. Under the event where theorem 2 holds (i.e. the event that occurs with probability at least $1-\delta-\delta'$), we had already shown that $R \leq \mathbb{E}[R] + Z_{Tm}$, for $Z_{Tm} \leq \sqrt{2Tk' \ln(1/\delta')}$, in the second part of the proof of theorem 2 . Thus when theorem 2 holds, we have $Z^2 \leq T + \sqrt{2Tk' \ln(1/\delta')} + \mathbb{E}[R]$. Now using the generalized version of weighted majority with rewards (i.e. using directly the benefits as rewards) (Arora et al., 2012), since the rewards at each update are in $[0, k']$, we have that with the best learning rate in hindsight [1]: $\mathbb{E}[R] \leq 2Z\sqrt{k' \ln|\tilde{\Pi}|}$. Thus we obtain $Z^2 \leq T + \sqrt{2Tk' \ln(1/\delta')} + 2Z\sqrt{k' \ln|\tilde{\Pi}|}$. This is a quadratic inequality of the form $Z^2 - 2Z\sqrt{k' \ln|\tilde{\Pi}|} - T - \sqrt{2Tk' \ln(1/\delta')} \leq 0$, with the additional constraint $Z \geq 0$. This implies Z is less than or equal to the largest non-negative root of the polynomial $Z^2 - 2Z\sqrt{k' \ln|\tilde{\Pi}|} - T - \sqrt{2Tk' \ln(1/\delta')}$. Solving for the roots, we obtain

$$\begin{aligned} Z &\leq \sqrt{k' \ln|\tilde{\Pi}|} + \sqrt{k' \ln|\tilde{\Pi}| + T + \sqrt{2Tk' \ln(1/\delta')}} \\ &\leq 2\sqrt{k' \ln|\tilde{\Pi}|} + \sqrt{T} + (2Tk' \ln(1/\delta'))^{1/4} \end{aligned}$$

[1] if not a doubling trick can be used to get the same regret bound within a small constant factor (Cesa-Bianchi et al., 1997)

Plugging back Z into the expression $\mathbb{E}[R] \leq 2Z\sqrt{k' \ln|\tilde{\Pi}|}$, we obtain:

$$\begin{aligned} \mathbb{E}[R] \leq \; & 4k' \ln|\tilde{\Pi}| + 2\sqrt{Tk' \ln|\tilde{\Pi}|} \\ & + 2(2T \ln(1/\delta'))^{1/4}(k')^{3/4}\sqrt{\ln|\tilde{\Pi}|} \end{aligned}$$

Thus the average regret:

$$\begin{aligned} \frac{\mathbb{E}[R]}{T} \leq \; & \frac{4k' \ln|\tilde{\Pi}|}{T} + 2\sqrt{\frac{k' \ln|\tilde{\Pi}|}{T}} \\ & + 2^{9/4}(k'/T)^{3/4}(\ln(1/\delta'))^{1/4}\sqrt{\ln|\tilde{\Pi}|} \end{aligned}$$

For T in $\Omega(k'(\ln\tilde{\Pi} + \ln(1/\delta')))$, the dominant term is $2\sqrt{\frac{k' \ln|\tilde{\Pi}|}{T}}$, and thus $\frac{\mathbb{E}[R]}{T}$ is $O(\sqrt{\frac{k' \ln|\tilde{\Pi}|}{T}})$. □

Corollary 3 . *Let $\alpha = \exp(-m/k)$ and $k' = \min(m, k)$. If we run an online learning algorithm on the sequence of convex loss C_t instead of ℓ_t, then after T iterations, for any $\delta \in (0,1)$, we have that with probability at least $1-\delta$:*

$$F(\overline{\pi}, m) \geq (1-\alpha)F(L^*_{\pi,k}) - \frac{\tilde{R}}{T} - 2\sqrt{\frac{2\ln(1/\delta)}{T}} - \mathcal{G}$$

where $\tilde{R}$ is the regret on the sequence of convex loss C_t, and $\mathcal{G} = \frac{1}{T}[\sum_{t=1}^{T}(\ell_t(\pi_t) - C_t(\pi_t)) + \min_{\pi \in \tilde{\Pi}} \sum_{t=1}^{T} C_t(\pi) - \min_{\pi' \in \tilde{\Pi}} \sum_{t=1}^{T} \ell_t(\pi')]$ is the "convex optimization gap" that measures how close the surrogate losses C_t is to minimizing the cost-sensitive losses ℓ_t.

Proof. Follows immediately from Theorem 2 using the definition of R, $\tilde{R}$ and $\mathcal{G}$, since $\mathcal{G} = \frac{R-\tilde{R}}{T}$ □

References

Arora, S., Hazan, E., , and Kale, S. The multiplicative weights update method: A meta-algorithm and applications. *Theory of Computing*, 8:121–164, 2012.

Cesa-Bianchi, N., Freund, Y., Haussler, D., Helmbold, D. P., Schapire, R. E., and Warmuth, M. K. How to use expert advice. *Journal of the ACM*, 44(3): 427–485, May 1997.

Feige, U., Mirrokni, V. S., and Vondrak, J. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

Streeter, Matthew and Golovin, Daniel. An online algorithm for maximizing submodular functions. Technical Report CMU-CS-07-171, Carnegie Mellon University, 2007.