Knapsack Constrained Contextual Submodular List Prediction with Application to Multi-document Summarization

Jiaji Zhou JIAJIZ@ANDREW.CMU.EDU
Stephane Ross STEPHANEROSS@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

We study the problem of predicting a *set* or *list* of options under knapsack constraint. The quality of such lists are evaluated by a submodular reward function that measures both quality and diversity. Similar to DAgger (Ross et al., 2010), by a reduction to online learning, we show how to adapt two sequence prediction models to imitate greedy maximization under knapsack constraint problems: CONSEQOPT (Dey et al., 2012a) and SCP (Ross et al., 2013). Experiments on extractive multi-document summarization show that our approach outperforms existing state-of-the-art methods.

1. Introduction

Many problem domains, ranging from web applications such as ad placement and content recommendation (Yue & Guestrin, 2011), to identifying successful robotic grasp trajectories (Dey et al., 2012a), to extractive multi-document summarization (Lin & Bilmes, 2010), 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 items) with maximal utility.

In all of these problems, the predicted list should be both relevant and diverse. For example, in extractive multi-document summarization, one must extract a small set of sentences (as a summary) to match human expert annotations (as measured via ROUGE (Lin, 2004) statistics). In this setting, selecting redundant sentences will not increase information coverage (and thus the ROUGE score). This notion of diminishing returns due to redundancy is often captured formally using submodularity (Guestrin & Krause).

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

However, in many settings such as document summarization, the reward function is only directly measurable on a finite training set (e.g., where we have expert annotations for computing the ROUGE score). As such it is increasingly common to take a supervised learning approach, where the goal is to learn a model or policy (based on the training set) that can make good predictions on new test examples where the reward function is not directly measurable.

Prior (state-of-the-art) work on document summarization (Lin & Bilmes, 2010; Kulesza & Taskar, 2011) first learn a surrogate submodular utility function that approximates the ROUGE score, and then perform approximate inference such as greedy using this surrogate function. While effective, such approaches are only indirectly learning to optimize the ROUGE score. For instance, small differences between the surrogate function and the ROUGE score may lead to the greedy algorithm performing very differently.

In contrast to prior work, we aim to directly learn to make good greedy predictions, i.e., by learning (on the training set) to mimic the clairvoyant greedy policy with direct access to the reward function. We consider two learning reduction approaches. Both approaches

Presented at the International Conference on Machine Learning (ICML) workshop on *Inferning: Interactions between Inference and Learning*, Atlanta, Georgia, USA, 2013. Copyright 2013 by the author(s).

decompose the joint learning task into a sequence of simpler learning tasks that mimic each iteration of the clairvoyant greedy forward selection strategy.

The first learning reduction approach decomposes the joint learning a set or list of predictions into a sequence of separate learning tasks (Streeter & Golovin, 2008; Radlinski et al., 2008; Streeter et al., 2009). In (Dey et al., 2012b), this strategy was extended to the contextual setting by a reduction to cost-sensitive classification.[1] In the second approach, (Ross et al., 2013) proposed learning one single policy that applies to each position in the list. Both approaches learn to maximize a submodular reward function under simple cardinality constraints, which is unsuitable for settings where different items exhibit different costs.[2]

In this paper, we extend both learning reduction approaches to knapsack constrained problems and provide algorithms with theoretical guarantees.[3] Empirical experiments on extractive document summarization show that our approach outperforms existing state-of-the-art methods.

2. Background

Let $S = \{s_1, \ldots, s_N\}$ denote a set of items, where each item s_i has length $\ell(s_i)$. Let $L_1 \subseteq S$ and $L_2 \subseteq S$ denote two sets or lists of items from S.[4] Let $\oplus$ denote the list concatenation operation.

We consider set-based reward functions $f : 2^{|S|} \rightarrow \mathbb{R}^+$ that obeys the following two properties:

1. **Submodularity:** for any two lists L_1, L_2 and any item s, $f(L_1 \oplus s) - f(L_1) \leq f(L_1 \oplus L_2 \oplus s) - f(L_1 \oplus L_2)$.
2. **Monotonicity:** for any two lists L_1, L_2, $f(L_1) \leq f(L_1 \oplus L_2)$ and $f(L_2) \leq f(L_1 \oplus L_2)$.

Intuitively, submodularity corresponds to a diminishing returns property and monotinicity indicates that adding more items never reduces the reward. We assume for simplicity that f takes values in [0,1], and in particular $f(\emptyset) = 0$.

We further enforce a knapsack constraint, i.e., that the computed list L must obey

$$\ell(L) = \sum_{s \in L} \ell(s) < W,$$

where W is a pre-specified budget. The knapsack constraint can be enforced by truncating[5] f when the budget is exceeded: $f(L_1 \oplus L_2) = f(L_1)$, if $\ell(L_1) < W$ and $\ell(L_1) + \ell(L_2) > W$. It is simple to show that this truncation preserves monotonicity and submodularity. Hence, adding any element or list that causes excess of budget would not increase the function value. We denote $b(s \mid L) = \frac{f(L \oplus s) - f(L)}{\ell(s)}$ as the unit/normalized marginal benefit of adding s to list L.

For the multi-document summarization application, S refers to the set of all sentences in a summarization task, and $\ell(s)$ refers to the byte length of sentence s. The reward function f is then the ROUGE Unigram Recall score, which can be easily shown to be monotone submodular (Lin & Bilmes, 2011).

3. Contextual Submodular Sequence Prediction

We assume to be given a collection of states $x_1, \ldots, x_T$, where each x_t is sampled i.i.d. from a common (unknown) distribution D. Each state x_t corresponds to a problem instance (e.g., a document summarization task) and is associated with observable features or context. We further assume that features describing partially contructed lists are also observable.

We consider learning a sequence of k policies $L_{\pi,k} = (\pi_1, \pi_2, ..., \pi_k)$ with the goal of applying them sequentially to predict a list L_t for x_t: policy π_i takes as input the features of x_t and $L_{t,i-1}$ and outputs an item s in S_t to append as the ith element after $L_{t,i-1}$. Therefore $L_{\pi,k}$ will produce a list $L_t = \{\pi_1(x_t, L_{t,0}), \pi_2(x_t, L_{t,1}), ..., \pi_k(x_t, L_{t,k-1})\}$.

We consider two cases. In the first case, each π_i is unique, and so we are learning a list of policies. In

[1] Essentially, each learning problem aims to build a policy that best predicts an item for the corresponding position in the list so as to maximize the expected marginal utility.

[2] In the document summarization setting, different sentences have different lengths.

[3] This is similar to the DAgger approach (Ross et al., 2011a;b; Ross & Bagnell, 2012) developed for sequential prediction problems like imitation learning and structured prediction. Our approach can be seen as a specialization of this technique for submodular list optimization, and ensures that we learn policies that pick good items under the distribution of list they construct. However, unlike prior work, our analysis leverages submodularity and leads to several modifications of that approach and improved guarantees with respect to the globally optimal list.

[4] Note that we refer to "lists" and "sets" interchangeably in this section. We often use "list" to convey an implicit notion of ordering (e.g., the order under which our model greedily chooses the list), but the reward function is computed over unordered sets.

[5] We allow the budget to be exceeded, and truncate the function value accordingly only in training. As for prediction, we always pick the element that ranks highest and fits into budget. Hence, the budget would not be exceeded.

Algorithm 1 Knapsack Constrained Submodular Contextual Policy Algorithm.

Input: policy class Π, budget length W.
Pick initial policy π_1
for $t = 1$ **to** T **do**
  Observe features of a sampled state $x_t \sim D$ and item set S_t
  Construct list L_t using π_t.
  Define $|L_t|$ new cost-sensitive classification examples $\{(v_{ti}, c_{ti}, w_{ti})\}_{i=1}^{|L_t|}$ where:

- v_{ti} is the feature vector of state x_t and list $L_{t,i-1}$
- c_{ti} is a cost vector such that $\forall s \in S_t : \; c_{ti}(s) = \max_{s' \in \mathcal{S}_\sqcup} b(s'|L_{t,i-1}, x_t) - b(s|L_{t,i-1}, x_t)$
- $w_i = [\prod_{j=i+1}^{|L_t|}(1 - \frac{\ell(s_{t,j})}{W})]\ell(s_{t,i})$ is the weight of this example

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

the second case, we enforce that each π_i is actually the same policy, and so we are learning just a single policy. We refer to π^t as the online learner's current policy when predicting for state x_t (which can be either a list of policies or a single policy depending on the algorithm). For both cases (described below), we show how to extend them to deal with knapsack constraints.

3.1. CONSEQOPT: Learning a sequence of (different) policies

CONSEQOPT(Dey et al., 2012a) learns a sequence of policies under cardinality constraint by reducing the learning problem to k separate supervised cost-sensitive classification problems in batch. We consider the knapsack constraint case and provide and error bound derived from regret bounds for online training.

3.2. SCP: Learning one single policy

SCP (Ross et al., 2013) learns one single policy that applies to each position for the list. The algorithm and theoretical analysis apply to cardinality constrained problems. Under the same online learning reduction framework (Ross et al., 2010), we extend the analysis and algorithm to the knapsack constraint setting.

3.3. Algorithm

Algorithm 1 shows our algorithm for both knapsack constrained CONSEQOPT and SCP. At each iteration, SCP/CONSEQOPT constructs a list L_t for state x_t using its current policy/list of policies. We get the observed benefit of each item in S_t at every position of the list L_t, organized as $|L_t|$ sets of cost sensitive classification examples $\{(v_{ti}, c_{ti}, w_{ti})\}_{i=1}^{|L_t|}$, each consisting of $|S_t|$ instances. These new examples are then used to update the policy. Note that the online learner's update operation (Update) is different for CONSEQOPT and SCP. CONSEQOPT has one online learner for each of its position-dependent policy and $\{(v_{ti}, c_{ti}, w_{ti})\}$ is used to update the ith online learner, while SCP would use all of $\{(v_{ti}, c_{ti}, w_{ti})\}_{i=1}^{|L_t|}$ to update a single online learner.

3.3.1. Reduction to Ranking

In the case of a finite policy class Π, one may leverage algorithms like Randomized Weighted Majority and update the distribution of policies in Π. However, achieving no-regret for a policy class that has infinite number of elements is generally intractable. As mentioned above, both learning reduction approaches reduce the problem to a better-studied learning problem, such as cost-sensitive classification.[6]

We use a reduction to ranking that penalizes ranking an item s above another better item s' by an amount proportional to their difference in cost. Essentially, for each cost-sensitive example (v, c, w), we generate $|\mathcal{S}|(|\mathcal{S}|-1)/2$ ranking examples, one 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 with feature weight h, this would be translated into a weighted hinge loss of the form: $w|c(s) - c(s')| \max(0, 1 - h^\top(v(s) - v(s'))\text{sign}(c(s) - c(s')))$. At prediction time we simply predict the item s^* with highest score, $s^* = \arg\max_{s \in \mathcal{S}} h^\top v(s)$.

This reduction is equivalent to the Weighted All Pairs reduction (Beygelzimer et al., 2005) except we directly transform the weighted 0-1 loss into a convex weighted hinge-loss upper bound – this is known as using a convex *surrogate* loss function. This reduction is often advantageous whenever it is easier to learn relative orderings rather than precise cost.

3.4. Theoretical Guarantees

We now present theoretical guarantees of Algorithm 1 relative to a randomization of an optimal policy $\tilde{L}_\pi^*$ that takes the following form. Let $L_\pi^* = (\pi_1^*, \ldots, \pi_W^*)$ denote an optimal deterministic policy list of size W. Let $\tilde{L}_\pi^*$ denote a randomization of L_π^* that gener-

[6] The reduction is implemented via the Update subroutine in Algorithm 1.

ates predictions L_t in the following way: Apply each $\pi_i^* \in L_\pi^*$ sequentially to x_t, and include the prediction $\pi_i^*(x_t, L_{t,i-1})$ picked by π_i^* with probability probability $p = 1/\ell(\pi_i^*(x_t, L_{t,i-1}))$, or otherwise discard. Thus, we have

$$L_{t,i} = \begin{cases} L_{t,i-1} \oplus \pi_i^*(x_t, L_{t,i-1}) & \text{w.p. } \frac{1}{\ell(\pi_i^*(x_t, L_{t,i-1}))} \\ L_{t,i-1} & \text{w.p. } 1 - \frac{1}{\ell(\pi_i^*(x_t, L_{t,i-1}))} \end{cases}$$

We can also think of each policy as having probability of being executed to be inversely proportional to the cost of the element it picks. Therefore, in expectation, each policy will add the corresponding normalized/unit benefit to the reward function value.

Ideally, we would like to prove theoretical guarantees relative to the actual deterministic optimal policy. However, $\tilde{L}_\pi^*$ can be intuitively seen as an average behavior of deterministic optimal policy. We defer an analysis comparing our approach to the deterministic optimal policy to future work.

We present learning reduction guarantees that relate performance on our actual submodular list optimization task to the regret of the corresponding online cost-sensitive classification task. Let $\{\ell_t\}_{t=1}^T$ denote a sequence of losses in the corresponding online learning problem, where $\ell_t : \Pi \to \mathbb{R}^+$ represents the loss of each policy π on the cost-sensitive classification examples $\{v_{ti}, c_{ti}, w_{ti}\}_{i=1}^{|L_t|}$ collected in Algorithm 1 for state x_t. The accumulated regret incurred by the online learning subroutine (UPDATE) is denoted by $R = \sum_{t=1}^T \ell_t(\pi^t) - \min_{\pi \in \Pi} \sum_{t=1}^T \ell_t(\pi)$.

Let $F(\pi) = \mathbb{E}_{x \sim D}[f_x(\pi(x))]$ denote the expected value of the lists constructed by π. Let $\hat{\pi} = \arg\max_{t \in \{1,2,\ldots,T\}} F(\pi^t)$ be the best policy found by the algorithm, and define the mixture distribution $\overline{\pi}$ over policies such that $F(\overline{\pi}) = \frac{1}{T} \sum_{t=1}^T F(\pi^t)$.

We will focus on showing good guarantees for $F(\overline{\pi})$, as $F(\hat{\pi}) \geq F(\overline{\pi})$. We now show that, in expectation, $\overline{\pi}$ (and thus $\hat{\pi}$) must construct lists with performance guarantees close to that of the greedy algorithm over policies in Π if a no-regret subroutine is used:

Theorem 1. *After T iterations, for any $\delta \in (0, 1)$, we have that with probability at least $1 - \delta$:*

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

Theorem 1 implies that the difference in reward between our learned policy and the (randomization of the) optimal policy is upper bounded by the regret of the online learning algorithm used in UPDATE divided by T, and a second term that shrinks as T grows.[7]

Running any no-regret online learning algorithm such as Randomized Weighted Majority (Littlestone & Warmuth, 1994) in UPDATE would guarantee

$$\frac{R}{T} = O\left(\frac{\sqrt{WgT \ln |\Pi|}}{T}\right)$$

for SCP and CONSEQOPT,[8] where g is the largest possible normalized/unit marginal benefit. Note that when applying this approach in a batch supervised learning setting, T also corresponds to the number of training examples.

3.4.1. Convex Surrogate Loss Functions

Note that we could also use an online algorithm that uses surrogate convex loss functions (e.g., ranking loss) for computational efficiency reasons when dealing with infinite policy classes. As in (Ross et al., 2013), we provide a general theoretical result that applies if the online algorithm is used on any convex upper bound of the cost-sensitive loss. An extra penalty term is introduced that relates the gap between the convex upper bound to the original cost-sensitive loss:

Corollary 1. *If we run an online learning algorithm on the sequence of convex losses C_t instead of ℓ_t, then after T iterations, for any $\delta \in (0, 1)$, we have that with probability at least $1 - \delta$:*

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

where $\tilde{R}$ is the regret on the sequence of convex losses 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 are to minimizing the cost-sensitive losses ℓ_t.

The gap $\mathcal{G}$ may often 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 predictor which models accurately all the costs or accurately ranks the items by a margin. Similarly,

[7] Note that we compare against $(1 - 1/e) F(\tilde{L}_\pi^*)$ because exact submodular optimization (with perfect knowledge of f) is intractable but forward greedy selection has a $(1 - 1/e)$ approximation guarantee.

[8] Naively, CONSEQOPT would have average regret that scales as $O\left(\frac{k(\sqrt{WgT \ln |\Pi|})}{T}\right)$, since we must run k separate online learners. However, similar to lemma 4 in (Streeter & Golovin, 2008) and corollary 2 in (Ross et al., 2013), it can be shown that the average regret is the same as SCP.

in cases where the problem is near-realizable we would expect $\mathcal{G}$ to be small. We emphasize that this convex optimization gap term is not specific to our particular scenario, but is (implicitly) always present whenever one attempts to optimize classification accuracy, e.g. the 0-1 loss, via convex optimization.[9] This result implies that whenever we use a good surrogate convex loss, then using a no-regret algorithm on this convex loss will lead to a policy that has a good approximation ratio to the optimal list of policies.

4. Application to Document Summarization

We apply our knapsack versions of the SCP and CONSEQOPT algorithms to an extractive multi-document summarization task. Here we construct summaries subject to a maximum budget of characters W by extracting sentences in the same order of occurrence as in the original document.

Following the experimental set up from previous work of (Lin & Bilmes, 2010) (which we call SubMod) and (Kulesza & Taskar, 2011) (which we call DPP), we use the datasets from the Document Understanding Conference (DUC) 2003 and 2004 (Task 2) (Dang, 2005). The data consists of clusters of documents, where each cluster 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 length is 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 learn a policy that optimizes the ROUGE-1R objective with respect to the reference summaries, which can be easily shown to be monotone submodular (Lin & Bilmes, 2011).

Intuitively, we want to predict sentences that are both short and capture a diverse set of important concepts in the target summaries. This is captured in our definition of cost using the difference of normalized benefit $c_{ti}(s) = b(s|L_{t,i-1}, x_t) - \max_{s' \in \mathcal{S}} b(s'|L_{t,i-1}, x_t)$. We use a reduction to ranking as described in Section 3.3.[10]

[9] For instance, when training a SVM in standard batch supervised learning, we would only expect that minimizing the hinge loss is a good surrogate for minimizing the 0-1 loss when the analogous convex optimization gap is small.

[10] We use Vowpal Wabbit (Langford et al., 2007) for online training and the parameters for online gradient descent are set as default.

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 statistics on the DUC 2004 test set

4.1. Feature Representation

The features for each state/document x_t are sentence-level features. 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. We use the same quality features as in (Kulesza & Taskar, 2011).

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 list. A variety of similarity features were considered, the simplest being average squared distance of tf-idf vectors. Performance varied little depending on the details of these 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 $L_{t,k} \cup s_i$, which is equivalent to the determinant of submatrix $\det(G_{L_{t,k} \cup s_i})$ of the Gram Matrix G, whose elements are pairwise TF-IDF vector inner products; 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 of the elements in $L_{t,k}$.

4.2. Results

Table 1 documents the performance (ROUGE unigram statistics) of knapsack constrained SCP and CONSEQOPT compared with SubMod and DPP (which are both state-of-the-art approaches). "Greedy (Oracle)" corresponds to the oracle used to train DPP, CONSEQOPT and SCP. This method directly optimizes the test ROUGE score and thus serves as an upper bound on this class of techniques. We observe that both SCP and CONSEQOPT outperform SubMod and DPP in terms of all three ROUGE Unigram statistics.

Acknowledgements

This research was supported by NSF NRI *Purposeful Prediction* and the Intel Science and Technology Center on Embedded Computing. We gratefully thank Martial Hebert for valuable discussions and Alex Kulesza for providing data and code.

References

Beygelzimer, A., Dani, V., Hayes, T., Langford, J., and Zadrozny, B. 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. Predicting contextual sequences via submodular function maximization. *CoRR*, abs/1202.2112, 2012a.

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

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

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

Langford, John, Li, Lihong, and Strehl, Alex. Vowpal Wabbit, 2007.

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

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

Littlestone, N. and Warmuth, M.K. 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.

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

Ross, S., Gordon, G.J., and Bagnell, J.A. No-Regret Reductions for Imitation Learning and Structured Prediction. *Arxiv preprint arXiv:1011.0686*, 2010.

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

Ross, S., Munoz, D., Bagnell, J. A., and Hebert, M. Learning message-passing inference machines for structured prediction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011b.

Ross, Stephane, Zhou, Jiaji, Yue, Yisong, Dey, Debadeepta, and Bagnell, J.A. Learning policies for contextual submodular prediction. In *ICML*, 2013.

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

Streeter, M., Golovin, D., and Krause, A. Online learning of assignments. In *NIPS*, 2009.

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

Appendix - Proofs of Theoretical Results

This appendix contains the proofs of theoretical results presented in this paper. We also encourage readers to refer to (Ross et al., 2013) and its supplementary materials for the proof of the cardinality constrained case.

Preliminaries

We begin by proving a number of lemmas about monotone submodular functions, which will be useful to prove our main results. Note that we refer to "lists" and "sets" interchangeably in this section. We often use "list" to convey an implicit notion of ordering (e.g., the order under which our model greedily chooses the list), but the reward function is computed over unordered sets.

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

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

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

Proof. For any two lists 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. □

Corollary 1. *Let $\mathcal{S}$ be a set and f be a monotone submodular function defined on a list of items from $\mathcal{S}$. Let $\tilde{B} = \{\tilde{b}_1, \ldots, \tilde{b}_{|B|}\}$ denote a stochastic list generated stochastically from the corresponding deterministic list B as follows:*

$$\forall i : \tilde{b}_i = \begin{cases} b_i & w.p.\ \frac{1}{\ell(b_i)} \\ \text{ø (empty)} & \text{otherwise} \end{cases}.$$

Then we have that:

$$\mathbb{E}[f(A \oplus \tilde{B})] - f(A) \leq |B|\mathbb{E}_{s\sim U(B)}\left[\frac{f(A \oplus s) - f(A)}{\ell(s)}\right],$$

where the first expectation is taken over then randomness of $\tilde{B}$, and $U(B)$ denotes the uniform distribution on items in B.

Proof.

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

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

Lemma 2. *Let $\mathcal{S}$, f, A, $\tilde{B}$, B, and $U(B)$ be defined as in Corollary 1. Let $\ell(A) = \sum_{i=1}^{|A|} \ell(a_i)$ denote the sum of length of each element a_i in A, and let A_j denote the list of the first j items in A. Define $\epsilon_j = \mathbb{E}_{s\sim U(B)}[\frac{f(A_{j-1} \oplus s) - f(A_{j-1})}{\ell(s)}] - \frac{f(A_j) - f(A_{j-1})}{\ell(a_j)}$ as the additive error term in competing with the average marginal normalized benefits of the items in B when picking the j^{th} item in A (which could be positive or negative). Then for $\alpha = \exp(-\ell(A)/|B|)$, we have*

$$f(A) \geq (1-\alpha)\,\mathbb{E}[f(\tilde{B})] - \sum_{i=1}^{|A|}\left[\prod_{j=i+1}^{|A|}\left(1 - \frac{\ell(a_j)}{|B|}\right)\right]\ell(a_i)\epsilon_i.$$

Proof. Using the monotonicity property of f and Corollary 1, we have that:

$$\begin{aligned} \mathbb{E}[f(\tilde{B})] - f(A) &\leq \mathbb{E}[f(A \oplus \tilde{B})] - f(A) \\ &\leq |B|\mathbb{E}_{s\sim U(B)}\left[\frac{f(A \oplus s) - f(A)}{\ell(s)}\right]. \end{aligned}$$

Define $\Delta_j = \mathbb{E}[f(\tilde{B})] - f(A_j)$. We have that:

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

Rearranging the terms yields

$$\begin{aligned} \frac{\ell(a_{j+1})}{|B|}\Delta_j &\leq \ell(a_{j+1})\epsilon_{j+1} + \Delta_j - \Delta_{j+1} \\ \Delta_{j+1} &\leq \left(1 - \frac{\ell(a_{j+1})}{|B|}\right)\Delta_j + \ell(a_{j+1})\epsilon_{j+1}. \end{aligned}$$

Recursively expanding, we get

$$\Delta_{|A|} \leq \prod_{i=1}^{|A|}\left(1-\frac{\ell(a_i)}{|B|}\right)\Delta_0 + \sum_{i=1}^{|A|}\prod_{j=i+1}^{|A|}\left(1-\frac{\ell(a_j)}{|B|}\right)\ell(a_i)\epsilon_i.$$

The term $\prod_{i=1}^{|A|}(1-\frac{\ell(a_i)}{|B|})$ is maximized when all $\ell(a_i)$ are equal, therefore $\prod_{i=1}^{|A|}(1-\frac{\ell(a_i)}{|B|}) \leq (1-\frac{\ell(A)}{|A||B|})^{|A|} \leq \exp(|A|\log(1-\frac{\ell(A)}{|A||B|})) \leq \exp(-|A|\frac{\ell(A)}{|A||B|}) = \alpha$. Rearranging the terms and using the definition of $\Delta_{|A|} = f(\tilde{B}) - f(A)$ and $\Delta_0 = f(\tilde{B})$ prove the statement. □

Proofs of Main Results

We now provide the proofs of the main results in this paper. We refer the reader to the notation defined in section 3 and 5 for the definitions of the various terms used.

Theorem 1. *After T iterations, for any $\delta \in (0,1)$, we have that with probability at least $1-\delta$:*

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

Proof.

$$\begin{aligned}
F(\bar{\pi}) &= \frac{1}{T}\sum_{t=1}^{T} F(\pi^t) \\
&= \mathbb{E}_{x\sim D}\left[\frac{1}{T}\sum_{t=1}^{T} f_x(\pi^t(x))\right] \\
&= \left(1-\frac{1}{e}\right)\mathbb{E}_{x\sim D}[f_x(\tilde{L}^*_\pi(x))] \\
&\quad - \mathbb{E}_{x\sim D}\left[\left(1-\frac{1}{e}\right)f_x(\tilde{L}^*_\pi(x)) - \frac{1}{T}\sum_{t=1}^{T} f_x(\pi^t(x))\right] \\
&= \left(1-\frac{1}{e}\right)\mathbb{E}_{x\sim D}[f_x(\tilde{L}^*_\pi(x))] \\
&\quad - \left(1-\frac{1}{e}\right)\frac{1}{T}\sum_{t=1}^{T}\left(f_{x_t}(\tilde{L}^*_\pi(x_t)) - f_{x_t}(L_t)\right) \\
&\quad - \frac{1}{T}\sum_{t=1}^{T} X_t,
\end{aligned}$$

where

$$\begin{aligned}
X_t = (1-1/e)\{&\mathbb{E}_{x\sim D}[f_x(\tilde{L}^*_\pi(x))] \\
&- f_{x_t}(\tilde{L}^*_\pi(x_t))\} - \{\mathbb{E}_{x\sim D}[f_x(\pi^t(x))] - f_{x_t}(L_t)\}.
\end{aligned}$$

Because each x_t is sampled 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 each X_t when conditioned on $\{X_\tau\}_{\tau=1}^{t-1}$ will 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 $Y_t = \sum_{i=1}^{t} X_i$ 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 $1-\delta$, $Y_T/T = \frac{1}{T}\sum_{t=1}^{T} X_t \leq 2\sqrt{\frac{2\ln(1/\delta)}{T}}$. Hence we have that with probability at least $1-\delta$:

$$\begin{aligned}
&F(\overline{\pi}) \\
\geq\ & (1-1/e)\mathbb{E}_{x\sim D}[f_x(\tilde{L}^*_\pi(x))] \\
& -[(1-1/e)\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(\tilde{L}^*_\pi(x_t)) \\
& -\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(L_t)] - 2\sqrt{\frac{2\ln(1/\delta)}{T}}
\end{aligned}$$

Let $s_{t,i}$ denote the ith element in L_t and $w_i = [\prod_{j=i+1}^{|L_t|}(1-\frac{\ell(s_{t,j}))}{k})]\ell(s_{t,i})$. From lemma 2, we have:

$$\begin{aligned}
&(1-1/e)\tfrac{1}{T}\textstyle\sum_{t=1}^{T} f_{x_t}(\tilde{L}^*_\pi(x_t)) - \tfrac{1}{T}\sum_{t=1}^{T} f_{x_t}(L_t) \\
\leq\ & \tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{i=1}^{|L_t|} w_i(\mathbb{E}_{\pi\sim U(L^*_\pi)}[\frac{f_{x_t}(L_{t,i-1}\oplus\pi(x_t,L_{t,i-1}))}{\ell(\pi(x_t,L_{t,i-1}))} \\
& -\tfrac{f_{x_t}(L_{t,i})-f_{x_t}(L_{t,i-1})}{\ell(s_{t,i})}) \\
=\ & \mathbb{E}_{\pi\sim U(L^*_\pi)}\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{i=1}^{|L_t|} w_i(\frac{f_{x_t}(L_{t,i-1}\oplus\pi(x_t,L_{t,i-1}))}{\ell(\pi(x_t,L_{t,i-1}))} \\
& -\tfrac{f_{x_t}(L_{t,i})-f_{x_t}(L_{t,i-1})}{\ell(s_{t,i})}) \\
\leq\ & \max_{\pi\in\Pi}\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{i=1}^{|L_t|} w_i(\frac{f_{x_t}(L_{t,i-1}\oplus\pi(x_t,L_{t,i-1}))}{\ell(\pi(x_t,L_{t,i-1}))} \\
& -\tfrac{f_{x_t}(L_{t,i})-f_{x_t}(L_{t,i-1})}{\ell(s_{t,i})}) \\
=\ & R/T
\end{aligned}$$

Hence combining with the previous result proves the theorem. □

Corollary 1. *If we run an online learning algorithm on the sequence of convex losses C_t instead of ℓ_t, then after T iterations, for any $\delta \in (0,1)$, we have that with probability at least $1-\delta$:*

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

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

Proof. Follows immediately from Theorem 1using the definition of R, $\tilde{R}$ and $\mathcal{G}$, since $\mathcal{G} = \frac{R-\tilde{R}}{T}$ □