Learning Sparse Combinatorial Representations via Two-stage Submodular Maximization

Eric Balkanski ERICBALKANSKI@G.HARVARD.EDU
Harvard University

Andreas Krause KRAUSEA@ETHZ.CH
ETH Zurich

Baharan Mirzasoleiman BAHARANM@INF.ETHZ.CH
ETH Zurich

Yaron Singer YARON@SEAS.HARVARD.EDU
Harvard University

Abstract

We consider the problem of learning sparse representations of data sets, where the goal is to reduce a data set in manner that optimizes multiple objectives. Motivated by applications of data summarization, we develop a new model which we refer to as the *two-stage submodular maximization problem*. This task can be viewed as a combinatorial analogue of representation learning problems such as dictionary learning and sparse regression. The two-stage problem strictly generalizes the problem of cardinality constrained submodular maximization, though the objective function is not submodular and the techniques for submodular maximization cannot be applied. We describe a continuous optimization method which achieves an approximation ratio which asymptotically approaches $1 - 1/e$. For instances where the asymptotics do not kick in, we design a local-search algorithm whose approximation ratio is arbitrarily close to $1/2$. We empirically demonstrate the effectiveness of our methods on two multi-objective data summarization tasks, where the goal is to construct summaries via sparse representative subsets w.r.t. to predefined objectives.

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

1. Introduction

In this paper, we consider the task of learning combinatorial representations of data. We are motivated by the following genre of multi-objective summarization tasks: Given a collection of articles (say all articles about ML on Wikipedia), as well as a set of subcategories (kernel methods, neural networks, etc.), pick a set of articles that are representative with respect to the whole corpus (field of ML). We study such problems through a novel model we call *two-stage submodular maximization*.

Submodularity is a natural diminishing returns condition which serves as a rich abstraction of combinatorial information measures such as entropy (RRNYI, 1961), mutual information (Guiau, 1977), coverage (Wolsey, 1982) etc. Much recent work in machine learning has explored submodular maximization as a natural abstraction for data summarization tasks (e.g., extractive summarization of documents, image collections, videos etc. (Lin and Bilmes, 2011; 2012; Tschiatschek et al., 2014)). In these applications, one typically designs (or learns) a submodular utility function f, which quantifies the representativeness $f(S)$ of a subset S of items (e.g, pictures, sentences) w.r.t. a large data set (image collection, document). Given a constraint on the size of the desired summary, the combinatorial problem of finding a summary S of maximum utility reduces to constrained submodular optimization, for which a wealth of efficient algorithms with strong theoretical guarantees have been developed.

In order to model the task of learning data representations, we depart from the classical (single-stage) setup, and consider a more general, two-stage task: We are

given multiple submodular objectives $f_1, \ldots, f_m$, and aim to select a set S of size at most l that can serve as a ground set that yields a high value for all objectives. That is, we aim to select S s.t. for each f_i, when the optimal subset $S_i \subseteq S$ of size at most k is selected, the sum over all $f_i(S_i)$ is maximized. Clearly, if $m = 1$, or if $l = k$, this problem reduces to standard cardinality constrained submodular maximization. The restrictions $k < l$ and $m > 1$ however allows modeling a much richer class of problems. In our multi-objective summarization task for example, for each subcategory i, we use a different submodular objective which quantifies the representativeness of a set of articles w.r.t. all articles in the subcategory. Here, the two-stage setup ensures that no single category dominates the overall summary.

Two-stage submodular maximization can also be viewed as a combinatorial analogue of representation learning tasks such as dictionary learning (Mairal et al., 2009; Zhou et al., 2009), (convolutional) auto encoders (Vincent et al., 2010), topic modeling (Maas et al., 2011) etc. Concretely, in dictionary learning, we are given a collection of signals (say images represented as vectors), and seek to select a basis, which allows to sparsely reconstruct each signal. Here, the task of sparse reconstruction is analogous to single-stage submodular maximization; dictionary learning is analogous to two-stage submodular maximization, where in the first stage the dictionary is selected, and in the second stage, it is used for multiple sparse reconstruction tasks (one for each signal).

Single-stage submodular maximization can be near-optimally solved using greedy techniques (Nemhauser et al., 1978), however the two-stage objective is not submodular (see Appendix A). In this paper we consider two approaches that yield provable guarantees for the two-stage submodular optimization problem:

- **Continuous optimization.** We begin by describing a general framework for solving two-stage submodular maximization problems via continuous optimization. This approach provides an approximation arbitrarily close to $1 - 1/e$ for sufficiently large values of k. At a high level, we relax the problem to a continuous program whose integral solutions identify with the discrete two-stage problem, solve the relaxation, and then round the solutions. Unlike standard relaxation methods for submodular optimization, constructing the relaxations involves interpreting fractional solutions as *correlated distributions*, and we design a *dependent rounding* technique s.t. variables corresponding to elements in the second stage are rounded in a manner that depends on elements rounded in the first stage.
- **Local-search.** For cases in which k is small, we develop a framework based on local-search which guarantees an approximation arbitrarily close to $1/2$. This guarantee dominates that of the continuous approach for small k. At a high level, we perform a local search by initializing a suboptimal solution and iteratively replacing elements that improve a potential function.

2. Two-Stage Submodular Maximization

To formally describe the two-stage problem, let $\mathcal{F} = \{f_1(\cdot), \cdots, f_m(\cdot)\}$ denote a class of m functions, each defined over $N = \{a_1, \ldots, a_n\}$, i.e., $f_j : 2^N \to \mathbb{R}$. The goal is to find a subset of size l whose subsets of size k maximize the sum over $\mathcal{F}$:

$$\max_{S:|S|\leq l} \sum_{j=1}^{m} \max_{\substack{T:T\subseteq S \\ |T|\leq k}} f_j(T).$$

We denote the objective value by $F(S)$, i.e., $F(S) := \sum_{j=1}^{m} \max_{T\subseteq S,|T|\leq k} f_j(T)$. The crucial underlying assumption is that the functions $\{f_j\}_{j=1}^m$ are all *submodular*, *normalized* ($f(\emptyset) = 0$) and *monotone* ($S \subseteq T$ implies $f(S) \leq f(T)$). A function $f : 2^N \to \mathbb{R}_+$ is submodular if $f(S \cup T) \leq f(S) + f(T) - f(S \cap T)$. Equivalently, a function is submodular if it has a natural diminishing returns property: for any $S \subseteq T \subseteq N$ and $a \in N \setminus T$ a function is submodular if $f_S(a) \geq f_T(a)$, where $f_S(a) = f(S \cup \{a\}) - f(S)$. We also note that our results extend to the case where the cardinality constraint in the second stage is a different k_i for each f_i. Fig. 1 depicts the two-stage problem.

2.1. Warm up: two-stage modular optimization

To gain some intuition about the problem, we can consider the case in which the underlying functions $\{f_i\}_{i=1}^m$ are *modular*. Recall that $f : 2^N \to \mathbb{R}$ is modular if the value of a set equals the sum of the values of its singletons, i.e., $f(S) = \sum_{a\in S} f(a)$. In the case the functions f_i are modular it is not difficult to show that the objective function of the two stage problem is actually *monotone submodular*. This implies that we can apply the seminal greedy algorithm, which at every step selects the element with the largest marginal contribution, and obtain a $1 - 1/e$ approximation to the optimal solution (Nemhauser et al., 1978).

Are constant factor approximation guarantees achievable for two-stage optimization of general monotone submodular functions?

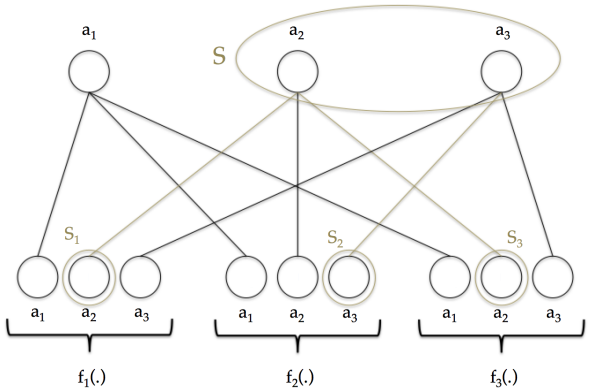


Figure 1. Example of the optimization problem with $N = \{a_1, a_2, a_3\}$, $m = 3$, $l = 2$, and $k = 1$.

2.2. General monotone submodular functions

When the underlying functions are not modular, the two-stage objective ceases to be submodular (see example in Appendix A) and we no longer have the approximation guarantees of the greedy algorithm. Slight relaxations of submodularity lead to strong inapproximability results, even for maximization under a cardinality constraint which is a degenerate case of our problem (Mirrokni et al., 2008). The main contribution of this paper is the general optimization frameworks we develop for the case in which the underlying functions are general monotone submodular.

3. Continuous Optimization

In this section, we describe a general technique which leads to an approximation that is arbitrarily close to the optimal $1 - 1/e$ approximation ratio (unless P=NP) for sufficiently large k. At a high level, we show that the two-level optimization problem can be solved through continuous optimization methods and a novel dependent rounding technique that may be of independent interest. In Appendix E, we extend this technique to obtain an approximation arbitrarily close to $1/e$ for non-monotone submodular functions.

A new ground set. Our formulation involves a ground set of size $n \times (m + 1)$ which includes the elements from the original ground set as well as an element for each possible $(element, function)$ pair. That is, our new ground set is $N' = N \cup \{a_{ij}\}_{i\in[n],j\in[m]}$. The unconstrained objective for the two stage problem over this new ground set is:

$$g(S) := \sum_{j=1}^{m} f_j(\{a_i \,:\, a_{ij} \in S\}).$$

The continuous problem. We associate each element from N' with a continuous variable $x_i \in [0, 1]$ for all a_i and $x_{ij} \in [0, 1]$ for all a_{ij}. We then aim to optimize:

$$\max G(\mathbf{x}) \tag{1}$$

$$\text{s.t.} \sum_i x_i \le l \tag{2}$$

$$\sum_i x_{ij} \le k \quad \forall j \in [m] \tag{3}$$

$$x_{ij} \le x_i \quad \forall (i, j) \in [n] \times [m] \tag{4}$$

$$x_i \le 1 \tag{5}$$

Constraints (2) and (3) correspond to the cardinality constraints for the first and second stages; (4) ensures that an element can only be picked in the second stage if it is picked in the first stage. The objective function $G(\mathbf{x})$ is the expected value of the integral solution when each element is picked with probability according to $\mathbf{x}$:

$$G(\mathbf{x}) := \mathbf{E}_{S\sim\mathcal{D}(\mathbf{x})}[g(S)]$$

where $\mathcal{D}(\mathbf{x})$ is a distribution with marginal probabilities $\mathbf{x}$ that we now define. To satisfy constraint (4), given $\mathbf{x} \in [0, 1]^{n+nm}$ we construct a *correlated* distribution $\mathcal{D}(\mathbf{x})$ with the following key properties[1]:

1. $a_i \in S \sim \mathcal{D}(\mathbf{x})$ for each i independently with probability x_i,
2. $a_{ij} \in S \sim \mathcal{D}(\mathbf{x})$ for each i and for each j independently with probability x_{ij}/x_i if $a_i \in S$ and with probability 0 otherwise.

We now discuss how we solve this continuous problem, which will return a *fractional* solution that results in the cardinality constraints only holding in expectation. We will later discuss the dependent randomized rounding scheme we develop, which ensures the constraints on every instance (albeit loses a fraction of the approximation guarantee that depends on k).

Optimiziation via continuous greedy. A seminal result by Vondrak shows that for any monotone submodular function, a polynomial-time continuous greedy algorithm can obtain an approximation arbitrarily close to $1 - 1/e$ of the optimal solution of the *multilinear extension* (ME). This guarantee holds when the solution $\mathbf{x}$ is constrained by a downward-closed solvable polytope (i.e. there is a poly-time separation oracle for the

[1]Note that if we pick elements independently, the constraint for the discrete problem that an element can only be picked in the second stage if it is picked in the first stage might be violated.

polytope) (Vondrák, 2008). The multilinear extension $G_{ME}(\mathbf{x})$ of a submodular function $g(\cdot)$ is the expected value of $g(S)$ when each element is picked ***independently*** with probability according to $\mathbf{x}$, i.e., $G_{ME}(\mathbf{x}) = \mathbf{E}_{S\sim\mathbf{x}}[g(S)]$. The crucial difference between the multilinear extension of the objective function of (1) is that the distribution is correlated. Despite this difference, we will use the continuous greedy algorithm to optimize program (1) and obtain a $1-1/e$ approximation, justified in two steps:

1. We show that the ME of $g(S)$ equals $G(\mathbf{x})$;
2. we observe that the constraints of the program (1) can be expressed as a downward-closed polytope.

With the distribution $\mathcal{D}(\mathbf{x})$ we constructed, the elements are picked dependently across functions and independently for a given function, so since $g(S)$ is additive over functions, the multilinear extension of $g(S)$ equals $G(x)$. The proof is deferred to Appendix B.

Lemma 3.1. *For any solution* $\mathbf{x} \in [0,1]^{n+nm}$*:*

$$G(\mathbf{x}) = G_{ME}(\mathbf{x}).$$

Now observe that program (1) can be equivalently reformulated over the same space of variables x_i and x_{ij}, but without any dependence on variables x_i, so that the feasible region is downward-closed.

$$\begin{aligned} \max\ & G(\mathbf{x}) \\ \text{s.t.}\ & \sum_i \max_j x_{ij} \le l \\ & \sum_i x_{ij} \le k \quad \forall j \in [m] \\ & x_{ij} \le 1 \quad \forall (i,j) \in [n]\times[m] \end{aligned} \tag{6}$$

Combining our previous observations, we can use continuous greedy to optimize program (6) and obtain a fractional solution that is a $1-1/e$ approximation to the optimal solution to the discrete problem.

Theorem 3.2. *Let* $\mathbf{x}$ *be the solution returned by the continuous greedy algorithm with program (6), and* $S^\star$ *be the optimal solution to the two stage problem. Then, for any* $\delta \ge poly(1/n)$*,*

$$G(\mathbf{x}) \ge (1-1/e-\delta)F(S^\star)$$

Proof. We know that continuous greedy returns a solution $\mathbf{x}$ that is a $1-1/e-\delta$ approximation to $G(\mathbf{x}^\star)$ where $\mathbf{x}^\star$ is the optimal solution for maximizing the multilinear extension of a submodular function under a downward-closed solvable polytope, for any δ that is not smaller than polynomial in $1/n$.[2] By Lemma 3.1, the objective of program (6) is the multilinear extension of $g(\cdot)$. The feasible region to program (6) is downward closed and solvable since there are polynomially many constraints.

It remains to show that $G(\mathbf{x}^\star) \ge F(S^\star)$, which we do by showing that the integral point $\mathbf{x}_{S^\star}$ corresponding to $S^\star$ is feasible. We formally define $\mathbf{x}_{S^\star}$ as $\mathbf{x}_{S^\star,i} = 1$ if $i \in S^\star$ and 0 otherwise, and $\mathbf{x}_{S^\star,ij} = 1$ if $i \in \arg\max_{T\subseteq S^\star,|T|\le k} f_j(T)$ and 0 otherwise. Since $\max_j \mathbf{x}_{S^\star,ij} = 1$ only if $\mathbf{x}_{S^\star,i} = 1$, the satisfiability of the constraints of program (6) then follow from $S^\star$ being feasible to the discrete two stage problem. □

Dependent rounding. We wish to ensure that the cardinality constraints not only hold in expectation but on every instance. At a high level, this is achieved by first solving the continuous problem for smaller values of l and k. We then construct a rounding scheme where each element is picked for each function with probability corresponding to the fractional solution and in a dependent way across functions for a fixed element. The solution obtained for this over-constrained problem satisfies the original constraints with high probability, and discarding instances where the original constraints do not hold only causes a small loss in the objective value. The formal construction and the analysis of this dependent rounding scheme use the framework of contention resolution schemes (Calinescu et al., 2011) discussed in Appendix C.

Theorem 3.3. *For any* $0 < \epsilon < 1/2$ *and* $\delta \ge poly(1/n)$*, the continuous optimization method results in a polynomial-time algorithm whose approximation ratio is* $1-1/e-1/k^{1/2-\epsilon}-2e^{-\Omega\left(k^{2\epsilon}\right)}-\delta$.

3.1. Optimization via LPs

The continuous greedy algorithm may be slow in practice. In Appendix D, we consider the case of coverage functions and formulate the continuous problem as a linear program, which can be solved substantially faster than continuous greedy. A function $f: 2^N \to \mathbb{R}$ is called coverage if there exists a family of sets $\{T_1,\ldots,T_n\}$ that are subsets of a universe U s.t. $f(S) = |\cup_{i\in S} T_i|$. Coverage functions are a special class of monotone submodular functions and often model diversity and representation objectives. In our case, selecting the most representative articles of a corpus is a special cases of two-stage maximization with coverage functions. The approximation ratio obtained with this linear programming approach is $1-1/e$ as well and the rounding is solved via the dependent rounding technique.

[2] Since $G(\cdot)$ cannot be evaluated in polynomial time, it is estimated by sampling, which causes an additional δ loss.

4. Local Search

We now describe the LOCAL-SEARCH algorithm, which provides an approximation arbitrarily close to $(e-1)/2e$ for coverage functions and $1/2$ for cases in which k is constant. For general k and general submodular functions, we slightly modify the algorithm in experiments with a heuristic which performs very well empirically.

The potential function. The main idea behind the algorithm is to (approximately) optimize a potential function through local search. The potential function is not submodular, yet has desirable properties amendable to provable guarantees. We begin by describing the approach for coverage functions with general parameter k and later define the approach for general submodular functions. For $\mathcal{F} = \{f_j\}_{j=1}^m$ the potential function is:

$$\Phi(S) = \sum_{j=1}^{m} \max_{\substack{\text{support}(\mathbf{x}) \subseteq S \\ \sum_i x_i \leq k}} L_j(\mathbf{x})$$

where L_j denotes the piecewise-linear relaxation of f_j:

$$L_j(\mathbf{x}) = \sum_{u \in U_j} \min\{1, \sum_{a \in C_j(u)} x_a\}$$

where $C_j(u)$ denotes the set of elements in the ground set that cover u from the universe U_j in coverage function f_j. Note that the piecewise linearity of L_j enables computing the optimal solution efficiently.

The local search algorithm. Algorithm 1 simply initializes a set S that includes the singleton with the largest contribution to the potential function together with $l-1$ arbitrary elements. For a given (fixed) precision parameter $\epsilon > 0$, the algorithm then iteratively replaces a single element in its current solution S if there is an element whose marginal contribution to the potential function is at least $1+\epsilon$ of the minimal marginal contribution in the current solution.

Algorithm 1 LOCAL-SEARCH

1: **input** constraints l, k, precision $\epsilon > 0$
2: $S \leftarrow \arg\max_{i \in N} \Phi(a_i) \cup$ arbitrary $l-1$ elements
3: **while** $\exists i \in S, j \notin S : (1-\epsilon)\Phi_S(a_j) > \Phi_{S \setminus a_i}(a_i)$ **do**
4: $\quad S \leftarrow (S \setminus \{a_i\}) \cup \{a_j\}$
5: **end while**
6: **return** S

Analysis of the algorithm. We first claim that the number of function evaluations is polynomial in the size of the problem (proof in Appendix F). We then bound the performance in terms of the potential function, which almost immediately gives the approximation guarantee.

Claim 4.1. *For any fixed $\epsilon > 0$ the* LOCAL-SEARCH *algorithm makes $O(k \cdot m \cdot l \cdot n^2 \log n)$ function evaluations.*

The following lemma shows that the marginal contribution of increasing some x_i^j to $L^j(\cdot)$ decreases as other $x_{i'}^j$ increase. We use the following notation: $\mathbf{x}^j = \arg\max_{\text{support}(\mathbf{x}) \subseteq S, \sum_i x_i \leq k} L_j(\mathbf{x})$, $\mathbf{x}^{j,i} := \{\mathbf{x}_1^j, \ldots, \mathbf{x}_i^j, 0, \ldots, 0\}$, and $\mathbf{x}_{-i}^j := \{\mathbf{x}_1^j, \ldots, \mathbf{x}_{i-1}^j, 0, \mathbf{x}_{i+1}^j \ldots, \mathbf{x}_n^j\}$. The proof is deferred to Appendix F.

Lemma 4.2. *The concave relaxation satisfies the following diminishing returns property:*

$$L_j(\mathbf{x}^{j,i}) - L_j(\mathbf{x}^{j,i-1}) \geq L_j(\mathbf{x}^j) - L_j(\mathbf{x}_{-i}^j).$$

Lemma 4.3. *Let S be the set returned by* LOCAL-SEARCH*, initialized with $\epsilon > 0$. Then:*

$$\Phi(S) \geq \left(\frac{1}{2} - O(\epsilon)\right) \max_{T:|T| \leq \ell} \Phi(T).$$

Proof. We first show that the potential is lower bounded by its marginals:

$$\begin{aligned}
\Phi(S) &= \sum_{j=1}^{m} L_j(\mathbf{x}^j) \\
&\geq \sum_{j=1}^{m} \sum_{i=1}^{n} L_j(\mathbf{x}^j) - L_j(\mathbf{x}_{-i}^j) \\
&\geq \sum_{j=1}^{m} \sum_{i=1}^{n} \max_{\substack{\text{support}(\mathbf{x}) \subseteq S \\ \sum_i x_i \leq k}} L_j(\mathbf{x}) - \max_{\substack{\text{support}(\mathbf{x}) \subseteq S \setminus a_i \\ \sum_i x_i \leq k}} L_j(\mathbf{x}) \\
&= \sum_{a_i \in S} \Phi_{S \setminus a_i}(a_i)
\end{aligned}$$

Now note that if S is the solution returned by LOCAL-SEARCH, this implies that no element $a_j \notin S$ can improve the solution. That is: $\Phi_S(a_j) \leq (1+\epsilon)\Phi_{S \setminus a_i}(a_i)$ for all $a_i \in S$ and $a_j \notin S$. Therefore, using $S^\star$ to denote the optimal solution to $\max_{T:|T| \leq \ell} \Phi(T)$ and combining with the previous observations, we get:

$$\begin{aligned}
\Phi(S) &\geq \sum_{a_i \in S} \Phi_{S \setminus a_i}(a_i) \\
&\geq \sum_{a_j \in S^\star} (1+\epsilon)\Phi_S(a_j) \\
&\geq (1-\epsilon)\Phi_S(S^\star) \\
&= (1-\epsilon)(\Phi(S \cup S^\star) - \Phi(S)) \\
&\geq (1-\epsilon)(\Phi(S^\star) - \Phi(S))
\end{aligned}$$

which concludes the proof. □

Theorem 4.4. *For any fixed $\epsilon > 0$ the* LOCAL-SEARCH *algorithm makes $O(k \cdot m \cdot \ell \cdot n^2 \log n)$ function evaluations and returns a set S that respects:*

$$F(S) \geq \left(\frac{1}{2}\left(1 - \frac{1}{e}\right) - O(\epsilon)\right) OPT$$

Proof. Let O be the optimal solution to the two-stage submodular maximization problem. It is well known that the concave relaxation of a coverage function upper bounds the cover function and is no more than a $1 - 1/e$ factor away from it, thus:

$$\begin{aligned} F(O) &\leq \Phi(O) \\ &\leq \max_{T:|T|\leq l} \Phi(T) \\ &\leq \left(\frac{1}{2} - \epsilon\right)^{-1} \Phi(S) \\ &\leq \left(\left(1 - \frac{1}{e}\right)\left(\frac{1}{2} - O(\epsilon)\right)\right)^{-1} F(S) \end{aligned}$$

where the third inequality follows from Lemma 4.3. □

General submodular functions. For general submodular functions we apply the local search method for small constant k when the approximation ratio from the previous section is strictly worse than $1/2$. We replace the potential function with the true objective $F(\cdot)$. For constant k, we can compute $F(S)$ by enumerating over all sets of size k for each function. We defer details and proofs to Appendix F. For large k one can compute a heuristic by computing the greedy solution instead of the optimal solution, as discussed in the following section.

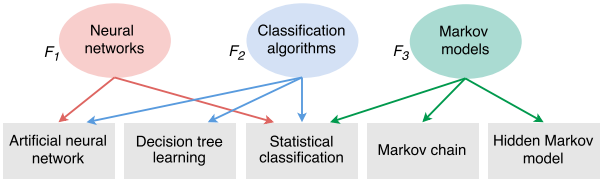
5. Experiments

We conduct data summarization experiments using two datasets: one of Wikipedia pages and one of images. The functions $f_i(\cdot)$ that we consider are coverage functions for the Wikipedia pages and more general submodular functions for the collection of images.

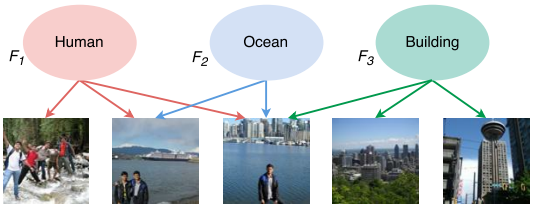
5.1. Datasets

Wikipedia pages. We apply our methods on the problem of picking a collection of articles that are most relevant for a set of diverse topics. We study the instance of such a problem where the articles are Wikipedia pages in a certain category and the different topics are subcategories of this category. The function $f_i(S)$ for each subcategory measures how relevant set S of Wikipedia pages is to subcategory i. More precisely, $f_i(S)$ is the number of Wikipedia pages that belong to subcategory i with a link to at least one page in S. Clearly, $f_i(S)$ is a cover function. For our experiments, the category of interest is machine learning, which contains $n = 575$ Wikipedia pages and has $m = 23$ subcategories. Fig. 2a shows an example of subcategories of machine learning, along with a solution of five pages for $l = 5$ and $k = 3$. Each subcategory points to its three most relevant pages.

Image collection summarization. In the image collection summarization problem, the goal is to select a small subset of images that best represents different categories. For example, one may have a collection of images taken on a holiday trip, and want to select a small subset that concisely represents all the diversity from the trip. Our experimental dataset is a collection of 100 images from Tschiatschek et al. (2014). We assigned each image to a subset of the following eight categories: human, building, tree, ocean, sky, mountain, road, and vehicle. Submodular function $f_i(S)$ indicates to what extent category i is represented by the summary set S. Formally, each $f_i(S)$ is a weighted linear combination of multiple submodular functions that capture different notions of representativeness. These functions are *Facility Location*, which measures the similarity of each image in category i to the closest image in S, *Sum Coverage* which measures the average similarity of each image in i to all images in S, *Truncated Graph Cut* which is similar to *Sum Coverage* but with some thresholding, as well as two functions rewarding diversity, *Clustered Facility Location*, which measures the similarity of each image in category i to the closest image from the same category in S, and *Clustered Diversity* that rewards selecting images from different categories. Details of all these functions can be found in (Tschiatschek et al., 2014). Fig. 2b shows three categories of images from an image collection, along with their most representative images from the collection.



(a) Wikipedia



(b) Images

Figure 2. Example of the most representative elements found by local search for constraints $l = 5$ and $k = 3$. The ovals show (sub)categories and below are their most relevant pages for (a) and their most representative images for (b).

5.2. Algorithms and baselines

LOCAL-SEARCH. We initialize LOCAL-SEARCH with the solution obtained by the greedy algorithm on the potential function $\hat{F}(S) = \sum_{i\in[m]} f_i(G_i(S,k))$, where $G_i(S,k)$ is the greedy solution with k elements for the second stage (i.e. not the optimal solution) w.r.t. f_i. To avoid enumerating over all subsets of size k, we executed a variant of the LOCAL-SEARCH algorithm for general submodular function where the potential function is evaluated with $\hat{F}(S)$ as above. In theory, substituting the optimal solution with the greedy algorithm in the evaluation of the potential function breaks the approximation guarantees. However, as we will see, this approach nonetheless is nearly optimal in practice. For the Wikipedia articles, we also run LOCAL-SEARCH with potential function $\phi(S)$ and initialized as above. We obtain near identical results as with the above approach.

CONTINUOUS-OPT. For CONTINUOUS-OPT we lowered the constraints k and l by a factor $1-\epsilon = 4/5$ for the rounding, which is an approximation of the optimal ϵ for these experiments.

Baselines. We compare our two algorithms, LOCAL-SEARCH and CONTINUOUS-OPT, to several natural baselines. Since continuous greedy is slow in practice, we applied CONTINUOUS-OPT only in the Wikipedia experiments with the linear programming approach for coverage functions. The baselines are natural variants of the greedy algorithm applied on various modifications of the objective functions that are submodular.

GREEDY-SUM first runs the greedy algorithm as if the cardinality constraint in the second stage was l, so it runs greedy on $F(S) = \sum_{j\in[m]} f_j(S)$ (which is a monotone submodular function) with cardinality constraint l. After a set S of l elements has been picked, we run the greedy algorithm over ground set S with cardinality constraint k for each function to obtain a feasible solution. **MODULAR-APPROX.** approximates the submodular functions in the second stage as if they were modular with each element having value $f_j(a)$. We then run the greedy algorithm on this approximation $F(S) = \sum_{j=1}^{m} \max_{T\subseteq S:|T|\leq k} \sum_{a\in T} f_j(a)$. Recall from our earlier discussion that if the functions $f_j(T)$ are modular, the two-stage problem is a special case of monotone submodular maximization under cardinality constraint. **CONCAVE-RELAXATION (upper bound)** is the value of the fractional solution with the concave relaxation objective for cover functions. This is an upper bound since we are able to compute the optimal solution to this concave relaxation which upper bounds the true objective. **GREEDY-MERGE (upper bound)** runs the greedy algorithm on each function in the second stage with a constraint of k. The solution may violate the constraint in the first stage. This upper bound can be applied to the images data set where the functions are not coverage.

5.3. Results

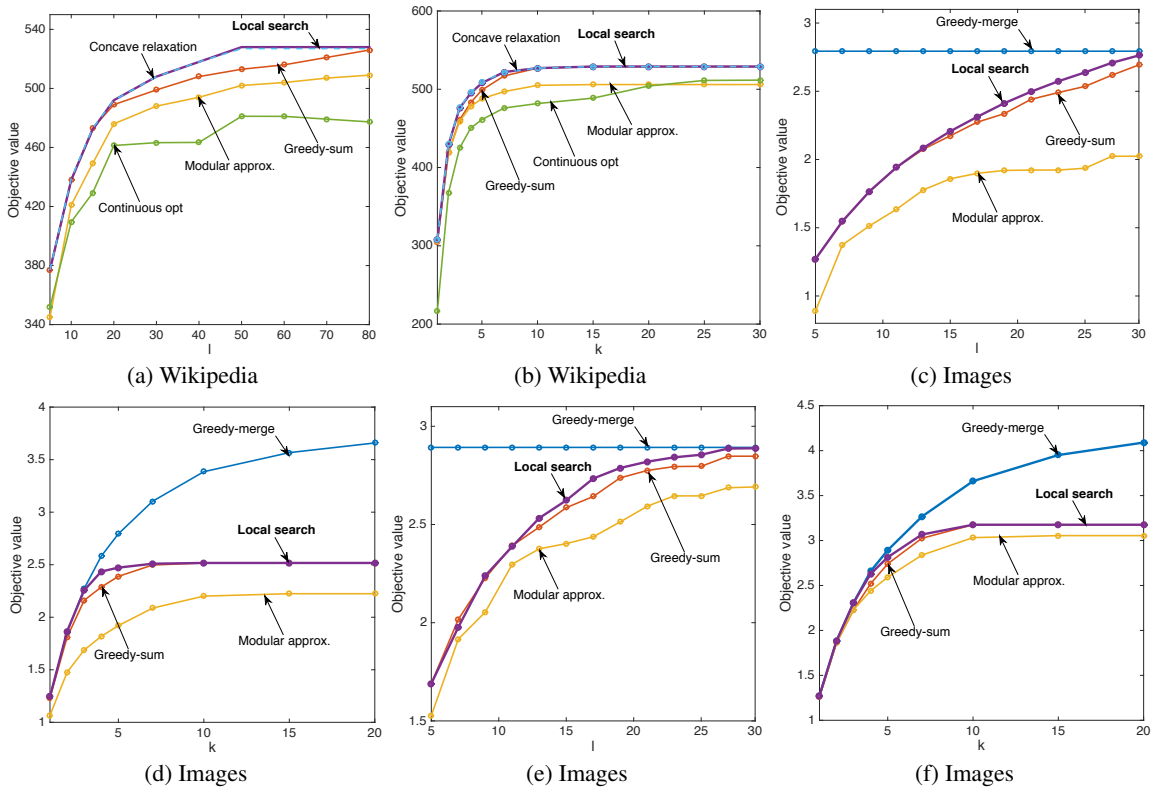
Fig. 3b and 3a show the performance of LOCAL-SEARCH and CONTINUOUS-OPT compared to the baselines for the Wikipedia dataset for varying l and k respectively. Fig. 3c and 3d are the results from the image dataset for varying l and k, where at most two categories from the categories listed previously are assigned to each image. Fig. 3e, and 3f show the same quantity when at most four categories are assigned to each image.

LOCAL-SEARCH is near optimal for the Wikipedia pages since there is very small gap compared to the upper bound CONCAVE-RELAXATION. It therefore performs much better in our experiments than its theoretical guarantee. It also outperforms all the baselines in each experiment. We also observed in the experiments that LOCAL-SEARCH with GREEDY initialization is fast: it requires fewer iterations than the theoretical upper bound given in Section 4. In fact, in most experiments, the solution returned by the GREEDY initialization is locally optimal and LOCAL-SEARCH does not perform any swaps. The performance of CONTINUOUS-OPT is dominated by that of LOCAL-SEARCH, and in most cases even dominated by the baselines. The theoretical guarantees for CONTINUOUS-OPT asymptotically improve as k and l grow large, the small values for l and k in our experiments cause a significant loss due to the rounding. The algorithms ranked in decreasing order of performance are LOCAL-SEARCH, GREEDY-SUM, MODULAR-APPROX., and CONTINUOUS-OPT. A further discussion of these experiments is in Appendix G.

6. Related Work

Submodular optimization has found numerous applications in machine learning and related fields, ranging from optimal sensor placement and variable selection in probabilistic models (Krause and Guestrin, 2005) to structure learning (**?**) to approximate inference in probabilistic models (Djolonga and Krause, 2014). In particular, submodular maximization has been found to be a natural abstraction of data summarization tasks (Lin and Bilmes, 2011; 2012; Tschiatschek et al., 2014; El-Arini et al., 2009). These approaches can be viewed as single-stage submodular maximization, which is strictly generalized by our approach.

Submodularity has been recently discovered to be relevant in tasks related to sparse reconstruction. For example, Bach (2010) shows how submodular functions can be used to define structured-sparse regularizers. Perhaps



(a) Wikipedia (b) Wikipedia (c) Images

(d) Images (e) Images (f) Images

Figure 3. Performance of LOCAL-SEARCH compared to the baselines. (a) shows the solution value for vertex cover on the Wikipedia dataset, for $k = 5$ and varying the solution size l, (b) shows the same quantity for $l = 30$, and varying k, (c) shows the solution value for summarizing a collection of 100 images according to the eight listed categories with $l = 20$ and varying k, (d) shows the same quantity for varying l and $k = 5$, (e) and (f) shows the same quantity for having additional categories per image for $l = 20$ and varying k, and for varying l and $k = 5$ respectively.

closest to our work is an approach of Cevher and Krause (2011); Das and Kempe (2011) on dictionary selection. Here, the goal is to select a collection of atoms (vectors in $\mathbb{R}^d$), which allow to sparsely represent a collection of signals. This problem is closely related to our two-stage maximization task, with the crucial difference that their individual objectives $f_1, \ldots, f_m$ quantifying reconstruction performance for the m signals are (approximately) *modular*. Hence their setting can be solved using classical submodular maximization (see Section 2.1).

There has also been recent significant interest in scaling submodular optimization to massive problems. For example, Mirzasoleiman et al. (2013) provided a simple two-stage distributed algorithm for submodular maximization under cardinality constraints. There have also been recent efforts to make use of stochastic methods to accelerate the running time of the centralized greedy algorithms (Mirzasoleiman et al., 2015). Streaming algorithms have also been proposed as another natural approach to scale up submodular optimization (Badanidiyuru et al., 2014). Wei et al. (2014); Kumar et al. (2013) have introduced multi-stage approaches. However, their goal is to accelerate performance, not to jointly optimize the performance over multiple stages as we do in this paper. Scaling our approach to massive problems using distributed/streaming computation is an exciting direction for future work.

7. Conclusions

In this paper, we have introduced a novel two-stage submodular optimization problem. This problem has natural applications in multi-objective summarization tasks, as we demonstrate in our experiments, and can be viewed as a novel combinatorial variant of representation learning tasks such as dictionary learning. We have presented two approaches: One based on continuous optimization, which handles general submodular functions, but is slow in practice, and another local search approach, which provides strong guarantees for special cases of the problem. Our experiments demonstrate the effectiveness and near-optimality of our approach.

Acknowledgements. This research was partially supported by ERC StG 307036, a Smith Family Graduate Science and Engineering Fellowship, NSF grant CCF-1301976, CAREER CCF-1452961.

References

Francis R Bach. Structured sparsity-inducing norms through submodular functions. In *Advances in Neural Information Processing Systems*, pages 118–126, 2010.

Ashwinkumar Badanidiyuru, Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Streaming submodular maximization: Massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 671–680. ACM, 2014.

Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

Volkan Cevher and Andreas Krause. Greedy dictionary selection for sparse representation. *Selected Topics in Signal Processing, IEEE Journal of*, 5(5):979–988, 2011.

Abhimanyu Das and David Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1057–1064, 2011.

Josip Djolonga and Andreas Krause. From map to marginals: Variational inference in bayesian submodular models. In *Neural Information Processing Systems (NIPS)*, December 2014.

Khalid El-Arini, Gaurav Veda, Dafna Shahaf, and Carlos Guestrin. Turning down the noise in the blogosphere. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 289–298. ACM, 2009.

Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 570–579. IEEE, 2011.

Silviu Guiasu. *Information Theory with New Applications*. McGraw-Hill Companies, 1977.

Andreas Krause and Carlos Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, page 5, 2005.

Ravi Kumar, Benjamin Moseley, Sergei Vassilvitskii, and Andrea Vattani. Fast greedy algorithms in mapreduce and streaming. In *Proceedings of the 25th ACM symposium on Parallelism in algorithms and architectures*, pages 1–10. ACM, 2013.

Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 510–520. Association for Computational Linguistics, 2011.

Hui Lin and Jeff A Bilmes. Learning mixtures of submodular shells with application to document summarization. *arXiv preprint arXiv:1210.4871*, 2012.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 142–150. Association for Computational Linguistics, 2011.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696. ACM, 2009.

Vahab Mirrokni, Michael Schapira, and Jan Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *Proceedings of the 9th ACM conference on Electronic commerce*, pages 70–77. ACM, 2008.

Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *Advances in Neural Information Processing Systems*, pages 2049–2057, 2013.

Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrak, and Andreas Krause. Lazier than lazy greedy. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294, 1978.

Franz Pernkopf and Jeff Bilmes. Discriminative versus generative parameter and structure learning of bayesian network classifiers. In *Proceedings of the 22nd international conference on Machine learning*, pages 657–664. ACM, 2005.

Alfréd Rényi. On measures of entropy and information. 1961.

Sebastian Tschiatschek, Rishabh K Iyer, Haochen Wei, and Jeff A Bilmes. Learning mixtures of submodular functions for image collection summarization. In *Advances in Neural Information Processing Systems*, pages 1413–1421, 2014.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010.

Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, pages 67–74, 2008.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. Fast multi-stage submodular maximization. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1494–1502, 2014.

Laurence A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 1982.

Mingyuan Zhou, Haojun Chen, Lu Ren, Guillermo Sapiro, Lawrence Carin, and John W Paisley. Non-parametric bayesian dictionary learning for sparse image representations. In *Advances in neural information processing systems*, pages 2295–2303, 2009.

Appendix

A. Two stage objective function is not submodular

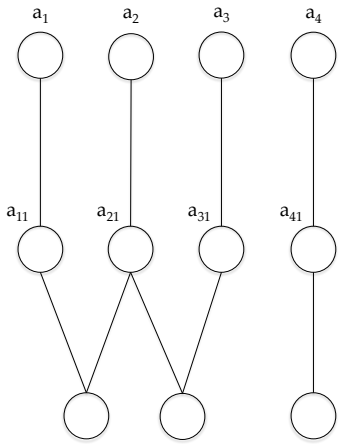


Figure 4. An example where the objective function for the two stage problem is not submodular. In this example, $f^1(\cdot)$ is a coverage function, $F_{\{a_1,a_3\}}(a_4) = 0$, and $F_{\{a_1,a_2,a_3\}}(a_4) = 1$ when $k = 2$.

Figure 4 is an example where the objective function for the two stage problem is not submodular. In this example, there is only one function $f^1(\cdot)$ in the second stage. The function $f^1(\cdot)$ is a coverage function defined as a bipartite graph where the value of a set of elements is the number of children that are covered by these elements in this bipartite graph. In Figure 4, the top level is the elements that can be picked in the first stage, the second level is the elements that can be picked in the second stage, and the graph between the second and third level is the bipartite graph defining the coverage function. We consider the case where $k = 2$. Observe that $F(\{a_1, a_3\}) = 2$ and that $F(\{a_1, a_3, a_4\}) = 2$ since at most 2 elements can be picked in the second stage and each of a_1, a_3, a_4 cover a unique child according to $f^1(\cdot)$, so $F_{\{a_1,a_3\}}(a_4) = 0$. Also observe that $F(\{a_1, a_2, a_3\}) = 2$ and that $F(\{a_1, a_2, a_3, a_4\}) = 3$ since $\{a_2, a_4\}$ cover 3 children, so $F_{\{a_1,a_2,a_3\}}(a_4) = 1$. The marginal return of adding a_4 to $\{a_1, a_3\}$ and $\{a_1, a_2, a_3\}$ is not diminishing so the function is not submodular.

B. Missing Lemma from continuous algorithm

Lemma. *For any solution* $\mathbf{x} \in [0, 1]^{n+nm}$*:*

$$G(\mathbf{x}) = G_{ME}(\mathbf{x}).$$

Proof. Let S_j be all the elements in a subset S picked for function $f_j(\cdot)$, i.e., $S_j := \{a_i \,:\, a_{ij} \in S\}$, then

$$\begin{aligned}
G(\mathbf{x}) &= \mathbf{E}_{S\sim\mathcal{D}(\mathbf{x})}[g(S)] \\
&= \sum_{S\subseteq N'} \mathbf{P}(S \sim \mathcal{D}(\mathbf{x})) \sum_{j=1}^{m} f_j(S_j) \\
&= \sum_{j=1}^{m} \sum_{T\subseteq N} \mathbf{P}(T = S_j \,:\, S \sim \mathcal{D}(\mathbf{x}))\, f_j(S_j) \\
&= \sum_{j=1}^{m} \sum_{T\subseteq N} \mathbf{P}(T = S_j \,:\, S \sim \mathbf{x})\, f_j(S_j) \\
&= \sum_{S\subseteq N'} \mathbf{P}(S \sim \mathbf{x}) \sum_{j=1}^{m} f_j(S_j) \\
&= \mathbf{E}_{S\sim\mathbf{x}}[g(S)] = G_{ME}(\mathbf{x})
\end{aligned}$$

where the fourth equality is justified by the fact that for a fixed j and when $S \sim \mathcal{D}(\mathbf{x})$, then the events $a_i \in S_j$ for each i are independent by definition of $\mathcal{D}(\mathbf{x})$. All the other equalities hold either by definition or by changing the order of the summations. □

C. Dependent rounding

We wish to have the constraints that not only hold in expectation but on every instance. At a high level, this is achieved by solving the continuous problem for smaller values of l and k. The fractional solution obtained for this over-constrained problem then satisfies the original constraints with high probability. We first review a known framework for this general rounding approach called contention resolution schemes (VCZ-11) and then construct a novel dependent rounding scheme using this framework.

Contention resolution schemes. Contention resolution schemes (VCZ-11) are a framework to obtain theoretical guarantees on rounding techniques for submodular optimization under different types of constraints. The setting is the following. Let I be the set of feasible integer solutions. We denote by P the polytope of fractional feasible solutions, which is also the convex relaxation of the constraints imposed by I. Scaling down P by a factor b is denoted by $bP = \{b \cdot x \,:\, x \in P\}$. A contention resolution scheme maps (both feasible and unfeasible) sets obtained from a fractional solution to feasible sets, it is defined formally as follows.

Definition 7.1 (VCZ-11 [3])**.** *A monotone (b, c) contention resolution scheme π for P is a deterministic procedure such that for every $x \in bP$ and $S \subseteq N$,*

- *(feasibility property) $\pi(S) \in I$ and $\pi(S) \subseteq S$,*
- *(marginal property) for all $i \in N$, $\mathbf{P}_{R\sim x}(i \in \pi(R) \,:\, i \in R) \geq c$, and*
- *(monotonicity property) for all $T \subseteq S$ and $i \in T$, if $i \in \pi(S)$, then $i \in \pi(T)$.*

A monotone (b, c) contention resolution scheme π is a rounding scheme with an additional loss of bc in a monotone submodular objective function. Scaling down P by a factor b causes a loss of at most b by submodularity. Rounding a fractional solution $x \in bP$ with π causes a loss of c by the following theorem.

Theorem 7.2 (BKNS-10, VCZ-11)**.** *Given a non-negative submodular function and a monotone (b, c) contention resolution scheme, then if $f(\cdot)$ is monotone,*

$$\mathbf{E}_{R\sim x}[f(\pi(R))] \geq c \cdot \mathbf{E}_{R\sim x}[f(R)]\,.$$

If $f(\cdot)$ is non monotone, there exists a pruning function η_f that can be evaluated in linear time such that for f for non-monotone,

$$\mathbf{E}_{R\sim x}[f(\eta_f(\pi(R)))] \geq c \cdot \mathbf{E}_{R\sim x}[f(R)]\,.$$

The following contention resolution scheme for cardinality constraints is obtained using classical concentration bounds.

Theorem 7.3 (VCZ-11 [4])**.** *Consider the monotone contention resolution scheme π for a cardinality constraint k such that $\pi(R) = R$ if $|R| \leq k$ and $\pi(R) = \emptyset$ otherwise. Given $\epsilon' \in (0, 1/2)$ and marginal probabilities x such that*

- *(scaled ex ante budget constraint) $\sum_i x_i \leq (1 - \epsilon')\, k$, and*
- *$k > 2/\epsilon'$,*

then, conditioned on any element being picked, the probability that the number of elements picked does not exceed the budget, i.e., $\mathbf{P}_{R\sim x}(|R| \leq k)$, is at least $1 - e^{-\epsilon'^2(1-\epsilon')k/12}$. Therefore π is a monotone $(1 - \epsilon', 1 - e^{-\epsilon'^2(1-\epsilon')k/12})$ for any cardinality constraint k where $\epsilon' > 2/k$.

[3] This definition of contention resolution schemes is a simplified version of the one in VCZ-11 that is sufficient for our purposes

[4] This theorem is simplified for our purposes.

The dependent rounding scheme. A naive approach could be to apply the above result for cardinality constraints to each function in the second stage independently. The issue that arises from this approach is that the constraint for the first stage might not be satisfied. In order to satisfy this constraint, we construct a *two stage dependent randomized rounding*. This rounding first picks a set of elements S for the first stage and then picks a subset S_j of S for each function in the second stage such that each element is picked for each function with probability according to the fractional solution. This rounding is *dependent* across variables corresponding to the same element in the second stage, however this is not an issue since we maintain independence across variables corresponding to the same submodular function. Independence across variables corresponding to a same submodular function is necessary to obtain a random set $R \sim x_j$ for each submodular function $f^j(\cdot)$.

Theorem 7.4. *For any $0 < \epsilon < 1/2$ and fractional solution $x = (x_{11}, x_{12}, \ldots, x_{nm})$, there exists a randomized dependent rounding scheme with a loss of $1 - 1/k^{1/2-\epsilon} - 2e^{-\Omega\left(k^{2\epsilon}\right)}$ in the objective function.*

Proof. We construct dependent contention resolution schemes π_j for each function $f^j(\cdot)$. Let x^* be the solution to the continuous problem with constraints k and l lowered by a factor of $(1-\epsilon')$ for some $\epsilon' > 2/k$. Let S be the random set where each element e_i is picked independently with probability $\max_j x^*_{ij}$. Let S_j be the random set where each element $e_i \in S$ is picked independently with probability $x^*_{ij}/\max_j x^*_{ij}$. Observe that S_j is a random set where each element $e_i \in N$ is picked independently with marginal probability $\max_j x^*_{ij} \cdot x^*_{ij}/\max_j x^*_{ij} = x^*_{ij}$. Then, the rounding scheme is simply $\pi_j(S_j) = S_j$ if both constraints l and k are not exceeded, and $\pi_j(S_j) = \emptyset$ otherwise.

We claim that the rounding schemes π_j are a $1 - \epsilon' - 2e^{-\frac{\epsilon'^2(1-\epsilon')k}{12}}$ contention resolution scheme for each coverage function j. We verify the three properties of contention resolution schemes. Clearly the feasibility property is satisfied, $\pi_j(S_j)$ satisfies budget k and $\pi_j(S_j) \subseteq S_j$. Regarding the marginal property, by Theorem 7.3, we get

$$\begin{aligned}
&\mathbf{P}_{S_j \sim x^*_{ij} \text{ for all } i}(i \in \pi(S_j) \,:\, i \in S_j)\\
=&\mathbf{P}(|S| \leq l \text{ and } |S_j| \leq k \,:\, i \in S_j)\\
=&\mathbf{P}(|S| \leq l : i \in S_j) \cdot \mathbf{P}(|S_j| \leq k : i \in S_j, |S| \leq l)\\
\geq&(1 - e^{-\epsilon'^2(1-\epsilon')k/12}) \cdot (1 - e^{-\epsilon'^2(1-\epsilon')k/12})\\
\geq&1 - 2 \cdot e^{-\epsilon'^2(1-\epsilon')k/12}
\end{aligned}$$

Finally, the monotone property holds trivially since a subset of a set that does not exceed a budget does not exceed this budget as well. Observe that this rounding does not only satisfy the constraints for each coverage function but also satisfies the constraints that the budget in the first stage is l and that an element can only be picked in the second stage if it is picked in the first stage. By combining the contention resolution scheme obtained and Theorem 7.2, the loss from rounding is

$$\begin{aligned}
&\sum_j \mathbf{E}_{S_j}\left[f^j(\pi_j(S_j))\right]\\
&\geq (1 - 2 \cdot e^{-\epsilon'^2(1-\epsilon')k/12}) \sum_j \mathbf{E}_{S_j}\left[f^j(S_j)\right].
\end{aligned}$$

Finally, since we lowered the constraints by a factor of $(1-\epsilon')$, the total loss in the objective is at most $(1-\epsilon') \cdot \left(1 - 2 \cdot e^{-\epsilon'^2(1-\epsilon')k/12}\right)$ by submodularity. Thus, with $\epsilon' = 1/k^{1/2-\epsilon}$ it is also at most $1 - 1/k^{1/2-\epsilon} - 2e^{-\Omega\left(k^{2\epsilon}\right)}$ for $0 < \epsilon < 1/2$. □

D. Linear program for coverage functions

The continuous greedy algorithm is slow in practice. We now consider the case of coverage functions for which we formulate the continuous problem as a linear program, which can be solved much faster than with continuous greedy. The rounding is then solved as previously with the two-stage dependent rounding. The m coverage functions are over universes $U_1, \cdots, U_m$. We call the elements in these universes *children*, not to be confound with the elements in N which we call *parents*. Each coverage function is a bipartite graph between children and parents where the value of a

set S of parents is the number of children that S covers, i.e., the number of children that are adjacent to at least one parent in S.

As previously, we have variables x_i and x_{ij} for all $i \in [n]$ and for all $i \in [m]$. We denote by $U := \{(j,u) : u \in U_j\}$ the collection of all children and by $C_{ju} := \{e_i : e_i \text{ adjacent to } u \text{ in } f^j(\cdot)\}$ the set of parents that cover child u in coverage function $f^j(\cdot)$. We obtain the following program.

$$\max_x \quad F(x) := \sum_{(j,u)\in U} \left(1 - \prod_{i\in C_{ju}} (1 - x_{ij})\right) \tag{7}$$

$$\text{s.t.} \quad \sum_{i\in N} x_i \le B \tag{8}$$

$$\sum_{i\in N} x_{ij} \le B_j \quad \forall j \tag{9}$$

$$x_i \ge x_{ij} \qquad \forall ij \tag{10}$$

$$0 \le x \le 1 \tag{11}$$

The term $\prod_{i\in C_{ju}}(1 - x_{ij})$ in the objective (7) is the probability that none of the parents of child $u \in U_j$ are picked. So the objective is the sum of the expected number of children covered in each coverage function. To obtain a linear program, we consider the following well-known concave relaxation program of the maximum coverage problem.

$$\max_x \quad L(x) := \sum_{(j,u)\in U} \min(1, \sum_{i\in C_{ju}} x_{ij})$$

$$\text{s.t.} \quad \sum_{i\in N} x_i \le B$$

$$\sum_{i\in N} x_{ij} \le B_j \quad \forall j$$

$$x_{ij} \le x_i \qquad \forall ij$$

$$0 \le x \le 1$$

Introducing variables $z_{j,u}$ corresponding to $\min(1, \sum_{i\in C_{ju}} x_{ij})$, we get a linear program.

$$\max_{x,z} \quad \sum_{(j,u)\in U} z_{j,u}$$

$$\text{s.t.} \quad z_{j,u} \le 1 \qquad \forall j,u \tag{12}$$

$$z_{j,u} \le \sum_{i\in C_{ju}} x_{ij} \quad \forall j,u \tag{13}$$

$$\sum_{i\in N} x_i \le B \tag{14}$$

$$\sum_{i\in N} x_{ij} \le B_j \quad \forall j \tag{15}$$

$$x_{ij} \le x_i \qquad \forall ij \tag{16}$$

$$0 \le x \le 1 \tag{17}$$

The following well-known result shows that optimizing the concave relaxation $L(x)$ gives a $1 - 1/e$ approximation for the continuous relaxation $F(x)$.

Lemma 7.5. *For all* $\mathbf{0} \le x \le \mathbf{1}$,

$$F(x) \ge \left(1 - \frac{1}{e}\right) L(x).$$

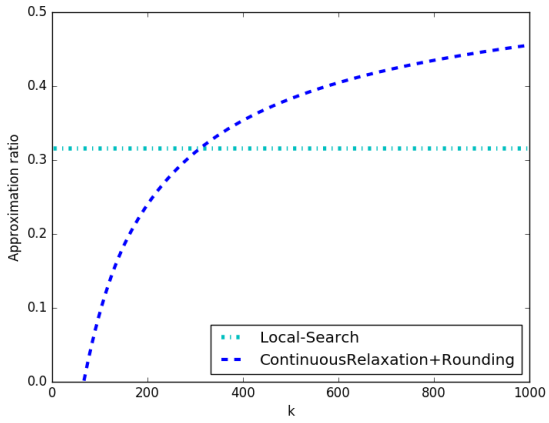


Figure 5. Approximation guarantees of the two algorithms as a function of the second stage constraint k.

E. Non-monotone submodular functions.

In the case of non-monotone submodular functions, the unified continuous greedy algorithm (**?**) obtains a $1/e - \delta$ approximation, for arbitrarily small δ, to the optimal solution of the ME under the same conditions as the continuous greedy algorithm. Since the analysis of the dependent rounding scheme also holds for non-monotone submodular functions, the approximation obtained by the continuous algorithm for monotone submodular functions extends to non-monotone submodular functions where the $1 - 1/e$ loss is replaced by $1/e$.

Theorem 7.6. *For any $0 < \epsilon < 1/2$ and $\delta \geq 1/poly(n)$, the continuous optimization method results in a polynomial-time algorithm whose approximation ratio for non-monotone submodular functions is $1/e - 1/k^{1/2-\epsilon} - 2e^{-\Omega\left(k^{2\epsilon}\right)} - \delta$.*

F. Missing analysis for LOCAL-SEARCH

Comparison with continuous algorithm. We compare the theoretical guarantees obtained by the two algorithms as a function of k in Figure 5. The approximation guarantee for the continuous algorithm is numerically optimized for the best ϵ for each k. We find that for $k \leq 311$, local search performs better than the continuous algorithm, otherwise the continuous algorithm performs best.

Claim. *For any fixed $\epsilon > 0$ the* LOCAL-SEARCH *algorithm makes $O(k \cdot m \cdot \ell \cdot n^2 \log n)$ function evaluations.*

Proof. Since the function $F(\cdot)$ is monotone and subadditive, the initialization step where we select the element with the largest value is guaranteed to be at least a factor of $1/n$ from the optimal solution. Using S^0 to denote the set selected and O as the optimal solution, since every iteration improves the approximation by a factor of at least $(1 + \epsilon)$ we have that:

$$\log_{1+\epsilon}\left(\frac{F(O)}{F(S^0)}\right) \leq \log_{1+\epsilon} n \in O(\log n)$$

In every iteration, the algorithm evaluates the potential marginal contribution of $n - \ell$ elements against ℓ elements in the current solution. Computing the marginal contribution requires running the greedy algorithm on each one of the m submodular functions $f^1, \ldots, f^m$. This takes, in the worst case, for each submodular function, running the greedy algorithm requires $k \cdot n$ function evaluations. Overall we have $O(k \cdot m \cdot \ell \cdot n^2 \log n)$. □

Lemma. *The concave relaxation satisfies the following diminishing returns property:*

$$L_j(\mathbf{x}^{j,i}) - L_j(\mathbf{x}^{j,i-1}) \geq L_j(\mathbf{x}^j) - L_j(\mathbf{x}^j_{-i}).$$

Proof. This property follows from the submodularity of coverage functions:

$$\begin{aligned}
&L_j(\mathbf{x}^{j,i}) - L_j(\mathbf{x}^{j,i-1}) \\
=&\sum_{u\in U_j} \min\{1, \sum_{e\in C_j(u)} x_e^{j,i}\} - \sum_{u\in U_j} \min\{1, \sum_{e\in C_j(u)} x_e^{j,i-1}\} \\
\geq&\sum_{u\in U_j} \min\{1, \sum_{e\in C_j(u)} x_e^{j}\} - \sum_{u\in U_j} \min\{1, \sum_{e\in C_j(u)} x_{-i,e}^{j}\} \\
=&L_j(\mathbf{x}^{j}) - L_j(\mathbf{x}^{j}_{-i})
\end{aligned}$$

where the inequality holds since the summations inside the minimum operator increases equally for both terms. □

LOCAL-SEARCH for general submodular functions. For general submodular functions our goal is to construct a method which obtains strong approximation guarantees for small values of k. To do so we use the local search method discussed above, where the potential function is simply $F(\cdot)$. That is, $\Phi(S) = F(S)$. The number of function evaluations is exactly the same as in the coverage case, and what remains is to argue about the approximation guarantee.

Theorem 7.7. *Let S be the set returned by* LOCAL-SEARCH*, initialized with $\epsilon > 0$. Then:*

$$F(S) \geq \left(\frac{1}{2} - O(\epsilon)\right) \max_{T:|T|\leq l} F(T).$$

Proof. Let $S^j = \{a_{j1}, \ldots, a_{jk}\} := \arg\max_{\substack{T\subseteq S \\ |T|\leq k}} f^j(T)$ and $S^j_i := \{a_{j1}, \ldots, a_{ji}\}$,

$$\begin{aligned}
F(S) = \sum_{j=1}^{m} f^j(S^j) &= \sum_{j=1}^{m}\sum_{i=1}^{k} f^j_{S^j_{i-1}}(a_{ji}) \\
&\geq \sum_{j=1}^{m}\sum_{i=1}^{k} f^j(S^j) - f^j(S^j \setminus a_{ji}) \\
&\geq \sum_{j=1}^{m}\sum_{i=1}^{k} \max_{\substack{T\subseteq S \\ |T|\leq k}} f^j(T) - \max_{\substack{T\subseteq S\setminus a_{ji} \\ |T|\leq k}} f^j(T) \\
&= \sum_{a_i\in S} F_{S\setminus a_i}(a_i)
\end{aligned}$$

where the first inequality is by submodularity and where the last equality is since the marginal contribution of an element not in S^j for f^j to S is 0. Similarly, we obtain that $\sum_{a_i\in S} F_T(a_i) \geq F_T(S)$.

Now note that if S is the solution returned by LOCAL-SEARCH, this implies that no element $a_j \notin S$ can improve the solution. That is: $F_S(a_j) \leq (1+\epsilon)F_{S\setminus a_i}(a_i)$ for all $a_i \in S$ and $a_j \notin S$. Therefore, using $S^\star$ to denote the optimal solution to $\max_{T:|T|\leq l} F(T)$ and combining with the previous observations, we get:

$$\begin{aligned}
F(S) &\geq \sum_{a_i\in S} F_{S\setminus a_i}(a_i) \\
&\geq \sum_{a_j\in S^\star} (1+\epsilon) F_S(a_j) \\
&\geq (1-\epsilon) F_S(S^\star) \\
&= (1-\epsilon)(F(S\cup S^\star) - F(S)) \\
&\geq (1-\epsilon)(F(S^\star) - F(S))
\end{aligned}$$

which concludes the proof. □

G. Further discussion of experiments

Wikipedia pages have non negligible value for more functions than images do for the image summarization experiments. The functions for the pages are near-additive due to the sparse structure of the network, which is not the case for the functions for images. Therefore, as a function of l, the plot for Fig. 3a looks concave and linear for Fig 3c. For Fig 3e where the images have non negligible values for more functions, the plot is less linear than for Fig 3c.

When $l \leq k$, the problem is submodular and GREEDY-SUM is equivalent to the classical greedy algorithm. When $l \geq km$, the problem is again submodular since each function can pick its k favorites elements, and the upper bound textscGreedy-merge returns a feasible solution. Thus, the further l is from k and km, the greater the differences in performance are (except with GREEDY-MERGE).