Fast Multi-Stage Submodular Maximization: Extended version

Kai Wei KAIWEI@U.WASHINGTON.EDU
Rishabh Iyer RKIYER@U.WASHINGTON.EDU
Jeff Bilmes BILMES@U.WASHINGTON.EDU
University of Washington, Seattle, WA 98195, USA

Abstract

Motivated by extremely large-scale machine learning problems, we introduce a new multi-stage algorithmic framework for submodular maximization (called MULTGREED), where at each stage we apply an approximate greedy procedure to maximize surrogate submodular functions. The surrogates serve as proxies for a target submodular function but require less memory and are easy to evaluate. We theoretically analyze the performance guarantee of the multi-stage framework and give examples on how to design instances of MULTGREED for a broad range of natural submodular functions. We show that MULTGREED performs very closely to the standard greedy algorithm given appropriate surrogate functions and argue how our framework can easily be integrated with distributive algorithms for further optimization. We complement our theory by empirically evaluating on several real-world problems, including data subset selection on millions of speech samples where MULTGREED yields at least a thousand times speedup and superior results over the state-of-the-art selection methods.

1 Introduction

Data sets are large and are getting larger. This, on the one hand, is useful as "there is no data like more data." On the other hand, it presents challenges since the information in vast quantities of data may be difficult to ascertain simply due to the computational difficulties created by vastness itself. An important goal in machine learning and information retrieval, therefore, is to develop methods that can efficiently extract and summarize relevant and useful information in large data sets.

One recent class of methods gaining some prominence in machine learning is based on submodular functions for combinatorial selection. Traditionally studied in mathematics, economics, and operations research, submodular functions naturally occur in many real world machine learning applications. A submodular function (Fujishige, 2005) is a discrete set function $f : 2^V \rightarrow \mathbb{R}$ that returns a real value for any subset $S \subseteq V$, and satisfies $f(A) + f(B) \geq f(A \cap B) + f(A \cup B), \forall A, B \subseteq V$. Equivalently, f satisfies the diminishing returns property, $f(j|S) \geq f(j|T), \forall S \subseteq T$, where $f(j|S) \triangleq f(j \cup S) - f(S)$ is the gain of adding element j a set S. A submodular function f is monotone non-decreasing if $f(j|S) \geq 0, \forall j \in V \setminus S, S \subseteq V$, and f is *normalized* if $f(\emptyset) = 0$.

Submodular functions naturally measure the amount of information that lies within a given set $A \subseteq V$, what is actually meant by "information" depends very much on the particular function. For example, given a collection of random variables $X_1, X_2, \ldots, X_n$, where $n = |V|$, the entropy function $f(A) = H(\cup_{a \in A} X_a)$ is submodular. The rank of a subset A of columns of a matrix is also submodular and can be seen as representing information as the dimensionality of the vector space spanned by the vectors indexed by A. There are other submodular functions that could represent "information" in some form (Fujishige, 2005; Kempe et al., 2003; Krause et al., 2008; Lin and Bilmes, 2011). Given a large collection of items V whose information content is $f(V)$ where f is submodular, a natural and common optimization problem is :

$$\max_{S \subseteq V, |S| \leq \ell} f(S). \tag{1}$$

Problem (1) asks for the most informative subset of items of size ℓ or less. This problem has already found great utility in a number of areas in machine learning, including document summarization (Lin and Bilmes, 2011), speech data subset selection (Wei et al., 2013), feature subset selection (Krause and Guestrin, 2005a; Liu et al., 2013), social influence (Kempe et al., 2003), and sensor placement (Krause et al., 2008). Though the problems are NP-hard, a well-known result by Nemhauser et al. (Nemhauser et al., 1978) shows that Problem 1 can be solved near-optimally by a simple greedy algorithm with a worst case approximation factor of $1 - 1/e$. Moreover, (Feige, 1998) shows that this is tight unless P=NP. In this paper, we shall refer to the solution of the greedy algorithm, though approximate, as the optimum solution for simplicity. The greedy algorithm starts with the empty

Proceedings of the 31^{st} International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

set $S_0 = \emptyset$. In each iteration i, it identifies the element s_i that maximizes the marginal gain $f(s_i|S_{i-1})$ (i.e., $s_i \in \operatorname{argmax}_{e \in V \setminus S_{i-1}} f(e|S_{i-1})$) with ties broken arbitrarily and updates as $S_i \leftarrow S_{i-1} \cup \{s_i\}$. Submodularity can be further exploited to accelerate (Minoux, 1978) this greedy algorithm – a procedure often called the "accelerated" or "lazy" greedy algorithm (LAZYGREED) (Leskovec et al., 2007).

In many cases, the very advanced machine learning algorithms that we need to use to process large data sources are too computationally costly for the amount of data that exists. For example, when the data source is very large (e.g., n in the billions or trillions), even LAZYGREED becomes untenable. Moreover, even smaller sized n can be prohibitive, particularly when evaluating the function f itself is expensive. For example, in document summarization (Lin and Bilmes, 2011) and speech data subset selection (Wei et al., 2013) certain submodular functions (which we call *graph-based*), are often defined via a pair-wise similarity graph, having a time and memory complexity of $O(n^2)$. This is infeasible even for medium-scale values of n. Another application is the feature selection (Krause and Guestrin, 2005a; Iyer and Bilmes, 2012; Liu et al., 2013), where a common objective is to maximize the mutual information between a given set of random variables X_A and a class C (i.e $I(X_A; C)$). The mutual information depends on computing the entropy $H(X_A)$ which can be expensive (or even exponential cost) to evaluate. Similarly, the recent promising work on determinantal point processes (DPPs) (Kulesza and Taskar, 2012), where one wants to maximize the submodular function $f(A) = \log\det(S_A)$ for a given matrix S, becomes problematic since computing log-determinant can require an $O(n^3)$ computation which is impractical already on medium-sized data sets.

We therefore ask the question: are there other approaches that can address Problems (1) and that are scalable to very large data sets, and that still offer theoretical guarantees?

Related Work: Parallel computing approaches are of course a natural pursuit for solving large-scale algorithmic challenges, and some instances of distributed algorithms for submodular optimization have already been investigated. For example, (Chierichetti et al., 2010) propose a distributed algorithm to solve Problem 1 with the set cover function as the objective, and with an approximation factor of $1-1/e-\epsilon$. Similarly, (Kumar et al., 2013) propose a distributed algorithm to solve Problem 1 with any submodular objective and with an approximation factor of $1/2 - \epsilon$. Motivated by the difficulty of rendering the entire data set centrally for function evaluation, (Mirzasoleiman et al., 2013) propose a two-stage algorithmic framework to solve Problem 1 with an approximation factor of $O(\frac{1}{\min(\ell,m)})$, where ℓ and m are the cardinality constraint and the number of distributed partitions, respectively. The performance guarantee can be improved to be close to optimum if the data set is massive and satisfies certain geometric assumptions. All these algorithms can be implemented in a Map-Reduce style.

Our Contributions: In this work, we propose a multi-stage framework (MULTGREED) that directly addresses the time and memory complexity issues of running LAZYGREED in three ways: (a) reducing the number of function evaluations required for the algorithm, (b) decreasing the complexity of function evaluations by using simpler surrogate (proxy) functions, (c) reducing the ground set size. Though quite different in spirit from the distributive framework, our approach could easily be performed in concert with existing distributed algorithms. For instance, we can apply MULTGREED instead of LAZYGREED for solving each sub-problem in (Mirzasoleiman et al., 2013). Conversely, their distributed procedure could also be used to solve the sub-problem in each stage of MULTGREED. The theoretical analysis for both frameworks could easily be combined with each other, and they could be integrated to provide still more efficient large-scale algorithmic frameworks for these problems. Hence, our approach is *complementary* to the existing distributive architectures, although in the present paper we will concentrate on our novel multi-stage uni-processor approach.

Outline Section 2 gives an overview of our framework. In Section 3, we theoretically analyze its performance while Section 4 offers several choices of surrogate functions for certain practical and useful classes of submodular functions. In Section 5, we focus on the design of MULTGREED on a broad range of submodular functions, and instantiate our general framework for several submodular functions, thereby providing recipes for many real-world problems. In Section 6, we empirically demonstrate the performance of MULTGREED, where we apply MULTGREED to a large-scale speech data subset selection problem and show that it yields superior results over the state-of-the-art selection methods.

2 Multi-Stage Algorithmic Framework

Often in applications, there is a desirable in quality but prohibitive in computational complexity submodular function that we shall refer to as the *target function* f. We assume a ground set size of $n = |V|$, a cardinality constraint of ℓ, and that the optimal solution to Problem (1) is S^{OPT}.

For completeness, we first describe how LAZYGREED accelerates the naive greedy implementation. The key insight is that the marginal gain of any element $v \in V$ is non-increasing during the greedy algorithm (a consequence of the submodularity of f). Instead of recomputing $f(v|S_{i-1})$ for each v, the accelerated greedy algorithm maintains a list of upper bounds $\rho(v)$ on each item's current marginal gain. They are initialized as $\rho(v) \leftarrow f(v)$ for each $v \in V$, and sorted in decreasing order (implemented as a priority queue). In iteration i, the algorithm pops the element v off the top of the priority queue and updates the bound $\rho(v) \leftarrow f(v|S_i)$. v is selected if $\rho(v) \geq \rho(u)$, where u is at the current top of the priority queue, since submodularity in such case guarantees that v provides the maximal marginal gain. Otherwise,

we appropriately place the updated $\rho(v)$ back in the priority queue and repeat.

To this end, we consider three schemes to further accelerate LAZYGREED: (a) reduce the number of function evaluations (Approximate greedy), (b) reduce the complexity of function evaluations (using simpler proxy functions), (c) reduce the ground set size (Pruning). This ultimately leads to our multi-stage greedy framework MULTGREED.

Approximate greedy: In this part, we introduce a mechanism called APPROXGREED, to reduce the number of function evaluations in LAZYGREED. We give a theoretical analysis for APPROXGREED in Section 3 — the current section defines and then offers intuition for the method. The key idea of APPROXGREED is that it does not insist on finding the item that attains exactly the maximum marginal gain in each iteration, but instead, looks for an item whose marginal gain is close to this maximum. APPROXGREED only modifies LAZYGREED by weakening the selection criteria in each iteration. More formally, if an item v is selected by LAZYGREED, the optimality of its marginal gain is guaranteed if the *exact condition* $\rho(v) \geq \rho(u)$ (u is the current top of the priority queue) is met. APPROXGREED relaxes this to an *approximate condition* $\rho(v) \geq \beta\rho(u)$, where $0 < \beta < 1$. Since a potentially large number of items' marginal gains need to be reevaluated until the exact condition is met, using the approximate condition could effectively reduce the number of function evaluations at a loss of the original guarantee. The parameter β controls the level of sub-optimality: the smaller β is, the number of function evaluations reduces as does the performance guarantee. In other words, APPROXGREED, as an approximate scheme to LAZYGREED, has its performance guarantee carried over from that of LAZYGREED, with an additional level of approximation governed by the value β (the formal guarantee of $(1 - e^{-\beta})$ is given in Lemma 2). We would like to point out the resemblance of APPROXGREED to the recently proposed fast greedy algorithm for Problem 1 (Badanidiyuru and Vondrák, 2014). Similar to APPROXGREED, they seek to identify an item whose marginal gain is within a fraction β of the maximum marginal gain in each iteration and yield the an approximation factor of $(1 - e^{-\beta})$. Unlike their algorithm, APPROXGREED builds on top of the LAZYGREED, hence, further exploits the submodularity. Though quite similar in spirit, APPROXGREED might run significantly faster, in practice, than their algorithm, while yielding the same performance guarantee.

APPROXGREED is only a minor modification of an existing LAZYGREED implementation and thus does not extend to other algorithms for Problem 1.

APPROXGREED can be further generalized by setting the value of β individually for each iteration, i.e., a sequence $\{\beta_i\}_{i=1}^{\ell} = \{\beta_1, \ldots, \beta_\ell\}$. Intuitively, we would design $\{\beta_i\}_{i=1}^{\ell}$ to be non-decreasing, i.e., the allowed sub-optimality decreases as the algorithm proceeds. The reason for this is that a less accurate selection at the beginning has a chance of being corrected by more accurate selection in later iterations. One possible schedule would be $\beta_i = c + \frac{1-c}{\ell}(i-1)$, where $c < 1$ determines the initial sub-optimality degree of the algorithm. Then, β_i grows linearly in i from c to 1, and the choice of c determines the trade-off between the running time reduction and performance guarantee loss. Given f, ℓ, and $\{\beta_i\}_{i=1}^{\ell}$, we shall instantiate the approximate greedy procedure as $S \in \text{APPROXGREED}(f, \ell, \{\beta_i\}_{i=1}^{\ell})$.

Multi-stage framework: APPROXGREED yields effective reduction on the number of function evaluations in LAZYGREED, however, the complexity of each function evaluation could still be so high that the greedy procedure is rendered impractical. To address this issue, we propose an approach, MULTGREED, that utilizes classes of simple surrogate functions which could be applied to a broad range of submodular functions. The idea is to optimize a series of surrogate (proxy) functions instead of optimizing the target function f.

Algorithm 1 MULTGREED: A Multi-Stage Greedy Alg.

Input f, ℓ, J, $\{f_j\}_{j=1}^{J}$, $\{\ell_j\}_{j=1}^{J}$, $\{\beta_i\}_{i=1}^{\ell}$
$C \leftarrow \emptyset$, $L \leftarrow 0$;
for $j = 1 \ldots J$ **do**
  Define $F_j(S) \triangleq f_j(S|C)$ for all $S \subseteq V$
  $S \in \text{APPROXGREED}(F_j, \ell_j, \{\beta_i\}_{i=L+1}^{L+\ell_j})$
  $L = L + \ell_j$, $C \leftarrow C \cup S$
Output C

Given a sequence $\{\beta_i\}_{i=1}^{\ell}$, a set of cardinality constraints $\{\ell_1, \ldots, \ell_J\}$ such that $\sum_{j=1}^{J} \ell_j = \ell$ and $\ell_j > 0, \forall j$, and a corresponding set of J surrogate (proxy) submodular functions $\{f_j\}_{j=1}^{J}$, we define our framework MULTGREED as shown in Algorithm 1. The series of the surrogate functions should be designed in increasing order of complexity, and at the last stage of MULTGREED, f_J can even be the target function f. The algorithm should typically start with a computationally simple surrogate submodular function f_1 (which could even be modular). Since the surrogate functions f_j's are designed to be computationally cheaper than the target function f, and since APPROXGREED is applied instead of LAZYGREED in each stage, we are guaranteed to achieve an overall reduction in computation. In practice (see Section 5 and 6), we often observe an instance of MULTGREED with $J = 2$ suffices to yield good enough performance and complexity reduction as well, though our results are much more general.

Pruning: In addition to the above two schemes, it is also desirable to prune out items of the ground set that will never be chosen anyway, especially for large-scale data set. This is commonly done for submodular *minimization* (Fujishige, 2005; Iyer et al., 2013b). Arbitrary pruning procedures, however, can significantly weaken the theoretical guarantee for Problem 1. We introduce here a simple new method that can prune away items without a corresponding performance loss. Consider the sequence of items $\{u_1, \ldots, u_n\}$ ordered

non-increasingly in terms of their gain conditioned on all other items, i.e., $f(u_1|V \setminus u_1) \geq \cdots \geq f(u_n|V \setminus u_n)$. For an instance of Problem 1 with cardinality constraint ℓ, we have the following Lemma:

Lemma 1. LAZYGREED *applied on the reduced ground set* $\hat{V} = \{j \in V | f(j) \geq f(u_\ell|V \setminus u_\ell)\}$ *is equivalent to that applied on the ground set* V.

The proofs for all the results in this paper are deferred to Appendix. This procedure can easily be implemented in parallel, since $f(j)$ and $f(j|V \setminus j)$ can be computed independently for all $j \in V$. The pruning procedure is optional and is applicable only when the complexity of evaluating $f(u|V \setminus u)$ is no greater than that of $f(u)$. This is the case, for example, in our graph-based submodular functions. It is not true, however, for the entropy-based functions, nor the log-determinant style functions. Otherwise, the complexity of the pruning procedure could potentially even dominate that of LAZYGREED, rendering it useless. MULTGREED may optionally start with this ground set pruning step, but it does not influence our analysis.

Our analysis of Algorithm 1 is given in Section 3, while in Section 4, we illustrate examples on how to design surrogate functions. In Section 5, we shall instantiate our framework and provide recipes for choosing the parameters of MULTGREED for several submodular functions which occur as models in real world applications.

3 Analysis

In this section, we formally analyze the methods presented in Section 2. We first define several crucial constructs that will facilitate this analysis.

Greedy ratio: We define a new construct we call the *greedy ratio* that will quantify the performance of a given instance of MULTGREED, which is characterized by the parameters: $\{f_j\}_{j=1}^J$, $\{\ell_j\}_{j=1}^J$, $\{\beta_i\}_{i=1}^\ell$. Guidelines on how to design the parameters of the multi-stage framework for several natural instances of useful submodular functions are given in Sections 4 and 5, but for now assume they are given. Let $s_1, \ldots, s_\ell$ be the sequence of items selected by the instance of MULTGREED. Let $S_i = \{s_1, \ldots, s_i\}$, be a set element of the chain $S_1 \subset S_2 \subset \cdots \subset S_\ell$, with $S_0 = \emptyset$.

Define the *individual greedy ratio* α_i for $i = 1, \ldots, \ell$ as:

$$\alpha_i = \frac{\max_{u \in V} f(u|S_{i-1})}{f(s_i|S_{i-1})} \tag{2}$$

Each α_i captures the ratio of the marginal gain of the greedily selected element to the marginal gain of the element s_i selected by MULTGREED. Therefore, α_i is a function of both the target function f but also, indirectly via ordered list $(s_1, s_2, \ldots, s_i)$, all of the remaining parameters $\{f_i\}_{i=1}^K$, $\{\ell_i\}_{i=1}^K$, $\{\beta_i\}_{i=1}^\ell$. Also, since $\max_{u \in V} f(u|S_{i-1}) \geq f(s_i|S_{i-1})$, we have that $\alpha_i \geq 1, \forall i$. Moreover, since under APPROXGREED we have $f(s_i|S_{i-1}) \geq \beta_i f(u|S_{i-1})$ for all $u \in V \setminus S_{i-1}$, it follows that $\alpha_i \leq 1/\beta_i$ for each i.

The list $\{\alpha_i\}_{i=1}^\ell$ collectively measures the quality of the multi-stage framework. We therefore define the *greedy ratio* α to be an aggregation of the individual greedy ratios. While there are many ways of aggregating, the harmonic mean, as we will show, provides the tightest characterization. We thus define the *greedy ratio* α as:

$$\alpha = \frac{\ell}{\sum_{i=1}^\ell 1/\alpha_i} \tag{3}$$

The greedy ratio, as we shall see, will provide a tight approximation guarantee. Ideally, we would like to have each individual greedy ratio $\alpha_i = 1$ for all i, and thus a greedy ratio of $\alpha = 1$. In particular, our strategy for choosing surrogate functions and other parameters is to induce a greedy ratio that is as small as possible.

Curvature: Another important construct we shall need is the curvature. Given a submodular function f, we define $\kappa_f(S)$ as the curvature of f with respect to a set S as follows: :

$$\kappa_f(S) = 1 - \min_{v \in V} \frac{f(v|S \setminus v)}{f(v)} \tag{4}$$

$\kappa_f(S)$ lies in the range of $[0, 1]$, and is monotonically non-decreasing in S. It measures the distance of f from modularity and $\kappa_f = 0$ if and only if f is modular (or additive, i.e., $f(S) = \sum_{i \in S} f(i)$). The total curvature (Conforti and Cornuejols, 1984) κ_f is then $\kappa_f(V)$. A number of approximation guarantees for submodular optimization are improved when using curvature (Conforti and Cornuejols, 1984; Iyer et al., 2013a; Iyer and Bilmes, 2013).

We now provide our main result:

Theorem 1. *Given a target submodular function* f *with total curvature* κ_f*, an instance of* MULTGREED *with greedy ratio* α *is guaranteed to obtain a set* S_ℓ *s.t.*

$$\frac{f(S_\ell)}{f(S^{OPT})} \geq \frac{1}{\kappa_f}\left(1 - \left(1 - \frac{1}{\alpha\ell}\right)^{\ell\kappa_f}\right) \geq \frac{1}{\kappa_f}\left(1 - e^{-\frac{\kappa_f}{\alpha}}\right) \geq \left(1 - e^{-\frac{1}{\alpha}}\right)$$

Conversely, for any value of $\alpha \geq 1$ *and* $\kappa_f \in [0, 1]$*, there exists a submodular* f *with the total curvature* κ_f*, on which an instance of* MULTGREED *with the greedy ratio* α *achieves an approximation factor* $\frac{1}{\kappa_f}\left(1 - \left(1 - \frac{1}{\alpha\ell}\right)^{\ell\kappa_f}\right)$.

Theorem 1 states that MULTGREED's guarantee is quantified tightly by the greedy ratio α. Moreover, the bound is, indirectly via α, dependent on all the parameters $\{f_i\}_{i=1}^K$, $\{\ell_i\}_{i=1}^K$, $\{\beta_i\}_{i=1}^\ell$ of MULTGREED. Theorem 1 generalizes bound $\frac{1}{\kappa_f}(1 - e^{-\kappa_f})$ (Conforti and Cornuejols, 1984) when $\alpha = 1$. By accounting for curvature, the bound $\frac{1}{\kappa_f}(1 - e^{-\kappa_f})$ itself generalizes the well-known result of $1 - 1/e$ for LAZYGREED on Problem 1. Also, as an immediate corollary of

Theorem 1, we obtain the theoretical guarantee for APPROXGREED in terms $\{\beta_i\}_{i=1}^{\ell}$.

Lemma 2. *Given a submodular function f with total curvature κ_f,* APPROXGREED$(f, \ell, \{\beta\}_{i=1}^{\ell})$ *is guaranteed to obtain a set S_ℓ: (here $\bar{\beta} = 1/\ell \sum_{i=1}^{\ell} \beta_i$)*

$$\frac{f(S_\ell)}{f(S^{OPT})} \geq \frac{1}{\kappa_f}(1 - (1 - \frac{\bar{\beta}}{\ell})^{\ell\kappa_f}) \geq \frac{1}{\kappa_f}(1 - e^{-\kappa_f \bar{\beta}}) \geq (1 - e^{-\bar{\beta}}),$$

If the $\{\beta_i\}_{i=1}^{\ell}$ are set as $\beta_i = c + \frac{1-c}{\ell} i$ with $0 \leq c \leq 1$ (c.f. Section 2), we have $\bar{\beta} \geq \frac{1+c}{2} \geq \frac{1}{2}$. Hence, this choice endows APPROXGREED with a solution having a factor no worse than $1 - e^{-1/2} \approx 0.39$.

The performance loss in MULTGREED comes from two sources, namely the approximate greedy procedure and the surrogate functions. To simplify our analysis, we henceforth utilize only the exact greedy algorithm, $(\forall i, \beta_i = 1)$. It should be clear, however, that our results will immediately generalize to the approximate greedy case as well.

The greedy ratio α is the harmonic mean of the values $\{\alpha_i\}_{i=1}^{\ell}$ that themselves can be partitioned into J blocks based on the J stages of MULTGREED. For $j = 1 \ldots J$, define $L_j = \sum_{j'=1}^{j} \ell_{j'}$, and let $I_j = \{L_{j-1} + 1, L_{j-1} + 2, \ldots, L_j\}$ be the set of ℓ_j indices for the j^{th} block. Each stage j provides a bound on the greedy ratio since $\alpha \leq \ell / \sum_{i \in I_j} 1/\alpha_i$. As a particularly simple example, if the target function f itself were to be utilized as the surrogate in the j^{th} stage for ℓ_j items, then each corresponding greedy ratio has value $\alpha_i = 1$ leading to the bound $\alpha \leq \ell/\ell_j$. Therefore, from the perspective of this upper bound, one is afforded the opportunity to design each stage semi-independently. On the other hand, to achieve a given desired α, it is not possible to design the stages entirely independently since the individual greedy ratios interact within the harmonic mean.

Generalization to knapsack constraint: Besides its flexibility in giving theoretical analysis of MULTGREED, the greedy ratio can work in a broader scenario. Consider a more general formulation of Problem 1 as:

$$\max_{c(S) \leq B, S \subseteq V} f(S) \tag{5}$$

where $c(S) = \sum_{v \in S} c(v)$ with $c(v) \geq 0$ being the cost of $v \in V$ and B is the budget constraint. Note many problems in machine learning applications, including sensor placement (Leskovec et al., 2007), document summarization (Lin and Bilmes, 2011), social networks (Singer, 2012) and training data subset selection (Wei et al., 2013), are formulated in this form. LAZYGREED for the cardinality constraint can be slightly modified to an algorithm, which we call *knapsack greedy algorithm*, to solve Problem 5 with factor $\frac{1}{2}(1 - e^{-1})$ (Krause and Guestrin, 2005b). Another variant of the greedy algorithm (Sviridenko, 2004) that achieves tight approximation factor $(1 - e^{-1})$ is not considered here, since its naive implementation requires $O(n^5)$ oracle access. The knapsack greedy algorithm differs from LAZYGREED in two aspects: (a) it, in each iteration, greedily selects the element that maximizes the marginal gain normalized by its cost, i.e., finding $s_i \in \operatorname{argmax}_{u \in V \setminus S_{i-1}} \frac{f(u|S_{i-1})}{c(u)}$; (b) it compares the final solution of the greedy algorithm with the maximum singleton value with cost under the budget constraint B, i.e., $\max_{e \in V, c(e) \leq B} f(e)$, and outputs the maximum of the two. The multi-stage framework designed to solve Problem 1 can be adapted to incorporate the above two modifications and applied to Problem 5. Assume an instance of the multi-stage procedure stops at iteration N, we denote the chain of items selected as $\{s_1, \ldots, s_N\}$, with $S_i = \{s_1, \ldots, s_i\}$ for $i = 1, \ldots, N$. We generalize the definition of the individual greedy ratio as:

$$\alpha_i = \frac{\max_{u \in V \setminus S_{i-1}} \frac{f(u|S_{i-1})}{c(u)}}{\frac{f(s_i|S_{i-1})}{c(s_i)}} \tag{6}$$

for $i = 1, \ldots, N$. And define the *knapsack greedy ratio* as

$$\alpha = \frac{B'}{\sum_{j=1}^{N} \frac{c(s_j)}{\alpha_j}} \tag{7}$$

where $B' = \sum_{i=1}^{N} c(s_i)$. It is worth pointing out the previously defined greedy ratio is a special case of the knapsack greedy ratio, where $c(v) = 1$ for $v \in V$.

Theorem 2. *Given a target submodular function f, an instance of* MULTGREED *with knapsack greedy ratio α is guaranteed to obtain a set S s.t.*

$$f(S) \geq \frac{1}{2}(1 - e^{-\frac{1}{\alpha}}) f(S^{OPT}) \tag{8}$$

where $S^{OPT} \in \operatorname{argmax}_{S \subseteq V, c(S) \leq B} f(S)$.

Theorem 2 generalizes the approximation factor $\frac{1}{2}(1 - e^{-1})$ (Krause and Guestrin, 2005b) where the knapsack greedy ratio is 1.

Generalization to submodular set cover problem: Closely related to Problem 5, the submodular set cover problem (Wolsey, 1982) is formulated as:

$$\min_{f(S) \geq f(V), S \subseteq V} c(S) \tag{9}$$

where f is a monotone submodular function and the same as before, $c(S) = \sum_{v \in S} c(v)$ with $c(v) \geq 0$ being the cost of item v. The same knapsack greedy algorithm solves Problem 9 with log factor approximation guarantee (Wolsey, 1982). Therefore, the same multi-stage greedy framework that solves Problem 5 can be carried over to solve Problem 9. Given the observation that Problem 5 and 9 are duals of each other in that solution for one problem can be efficiently transformed to a solution for the other with bi-criterion approximation guarantee (Iyer and Bilmes, 2013), we obtain the following result.

Theorem 3. *An instance of* MULTGREED *that solves Problem 5 with the knapsack greedy ratio α returns a solution that can be transformed to a solution S for Problem 9 such that*

$$c(S) \leq c(S^{OPT}) \text{ and } f(S) \geq \frac{1}{2}(1 - e^{-\frac{1}{\alpha}})f(V), \quad (10)$$

where $S^{OPT} \in \operatorname{argmax}_{S \subseteq V, f(S) \geq f(V)} c(S)$

The solution only satisfies approximate feasibility to Problem 9. Theorem 3 shows that an instance of the multi-stage knapsack greedy algorithm provides constant factor bi-criterion approximation guarantee, although Problem 9 does not admit any constant-factor approximation algorithms (Feige, 1998).

4 Surrogate Functions

In this section, we investigate the interplay between the greedy ratio and several choices of surrogate functions for classes of submodular functions which appear often in practice. Since providing bounds on the performance of each stage individually implies an upper bound on the greedy ratio, we shall restrict ourselves to the analysis of the surrogate function at a given stage j, and the final performance guarantee is easily obtained by combining the guarantees for the different stages.

Uniform Submodular Mixtures: We first consider a class of submodular functions that can be represented as

$$f(S) = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} f_t(S), \quad (11)$$

where $|\mathcal{T}| > 1$, and f_t is monotone submodular $\forall t \in \mathcal{T}$. We name this class *uniform submodular mixtures* as they are similar to the submodular mixtures previously defined in the context of learning (Lin and Bilmes, 2012). They are also similar to the *decomposable* submodular functions of (Stobbe and Krause, 2010) but without the requirement that $f_t(S)$ be a non-decreasing concave function composed with a non-negative modular function. A number of natural submodular functions belong to this class.

The complexity of evaluating f is determined both by $|\mathcal{T}|$ and the complexity of evaluating individual f_t's. Given such an f, a natural class of random surrogates takes the form

$$\hat{f}^{\text{sub}}(S) = \frac{1}{|\mathcal{T}'|} \sum_{i \in \mathcal{T}'} f_t(S), \quad (12)$$

where $\mathcal{T}' \subseteq \mathcal{T}$, and $\mathcal{T}'$ is generated by sampling individual elements from $\mathcal{T}$ with probability p. As p decreases, so does the complexity of evaluating $\hat{f}^{\text{sub}}$ but at the cost of a worse approximation to f. Applying a random function $\hat{f}^{\text{sub}}$ derived in this way to MULTGREED, and assuming $|f_t(S)| \leq \mathcal{B}, \forall t \in \mathcal{T}, S \subseteq V$, we obtain:

Lemma 3. *Using the surrogate uniform mixture $\hat{f}^{sub}$ for stage j in* MULTGREED *gives individual greedy ratios of*

$$1 \leq \alpha_i \leq \frac{1}{1-\epsilon}, \forall i \in I_j, \quad (13)$$

with probability $1-\delta$, where $\delta = (1 - 5n^{\ell} e^{-\frac{np(g^{\ell})^2\epsilon^2}{64\mathcal{B}^2}})$ and $g^{\ell} = \max_{u \in V \setminus S_{\ell-1}} f(u|S_{\ell-1}) > 0$.

Fixing δ, a smaller value of probability p yields a higher value of ϵ, weakening the bound on each α_i. Fixing both δ and ϵ, increases in the ground set size $n = |V|$ could yield a choice of surrogate function $\hat{f}^{\text{sub}}$ having a smaller sampling probability p and thus that is easier to evaluate. More importantly, fixing δ and p, ϵ can be made arbitrarily close to 0 for n sufficiently large, a result that is of great interest for very large-scale problems. We shall use this result to provide bounds for several instances of submodular functions in Section 5

Modular Upper bounds: We next focus on a class of surrogate functions applicable to general submodular functions. Given a submodular function f, its simple modular upper bound is given as

$$\hat{f}^{\text{mod}}(S) = \sum_{s \in S} f(s). \quad (14)$$

For some submodular functions such as entropy (including Gaussian entropy and the $\log\det$ functions used for DPPs) or mutual information, evaluating $\hat{f}^{\text{mod}}(S)$ is very easy, while evaluating $f(S)$ might sometimes even require computation exponential in $|S|$. Though extremely simple, this class nevertheless can act as an efficient class of surrogate functions especially useful when the target function is not very curved. $\hat{f}^{\text{mod}}$ is not only easy to optimize exactly, but it has previously been considered as a surrogate for various other forms of submodular optimization curvaturesubmodular, rkiyersemiframework2013, rishabh2013-submodular-constraints, rkiyersubmodBregman2012. The curvature κ_f, by definition, measures how close f is to being modular. If a modular surrogate function $\hat{f}^{\text{mod}}$, for general submodular function f, is applied within MULTGREED, we can thus bound the individual greedy ratio via the curvature:

Lemma 4. *Using the modular upper bound as a surrogate function, it holds that*

$$1 \leq \alpha_i \leq \frac{1}{1 - \kappa_f(S_{i-1})}, \forall i \in I_j$$

Unsurprisingly, we see that the less curved the target function f is, the tighter bound on α_i's, and the better $\hat{f}^{\text{mod}}$ performs as a surrogate. In particular, if f is modular, i.e., $\kappa_f = 0$, then, all individual greedy ratio α_i's are tightly bounded as 1. Lemma 4 also implies that the bound of the individual greedy ratio weakens as i increases, since

$\frac{1}{1-\kappa(S_{i-1})}$ increases with i. Therefore, this modular proxy, if applied, is best done in the first (or at most early) stages of MULTGREED.

Graph based Submodular functions: We focus next on a class of submodular functions based on an underlying weighted graph and hence called *graph-based*. Many submodular functions used in machine learning applications belong to this class (Kolmogorov and Zabih, 2004; Wei et al., 2013; Liu et al., 2013; Lin and Bilmes, 2011). These functions require $O(n^2)$ time to compute and store, which is not feasible for large n.

To form surrogates for the class of graph-based submodular functions, a natural choice is to utilize spanning subgraphs of the original graph. One choice is the k-nearest neighbor graph (k-NNG), defined as the spanning subgraph formed with each vertex $v \in V$ connected only to its k most similar neighbors (under the similarity score given by the edge weights). k-NNG has found great utilities in many machine learning applications (Shah et al., 2011; Boiman et al., 2008). We write $\hat{f}^{\text{k-NNG}}$ as the surrogate function defined on a k-NNG for a graph-based submodular function f. The sparsity of the k-NNG depends on the value of k. The denser the graph (higher k), the costlier both the function evaluations and the memory complexity becomes. In Section 5, surprisingly we will show that $\hat{f}^{\text{k-NNG}}$, even for k as sparse as $O(\log n)$, can be good enough for certain graph-based functions.

5 Instantiations with Real World Submodular functions

Given the previously defined machinery to analyze MULTGREED, we now focus on a broad range of submodular functions that appear as models in real world applications, and provide guidelines on how to design the surrogate functions as well as how to choose the size constraints. We investigate the following special cases: 1) the facility location function, 2) saturated coverage function, 3) feature based function, 4) the set cover function. We focus on analyzing the theoretical guarantees for these functions here and demonstrate the performance of some of these empirically in the next section.

Facility location function: Given a weighted graph $G = (V, E)$, with $w_{u,v}$ the edge weight (i.e., similarity score) between vertices u and v for $u, v \in V$, the (uncapacitated) facility location function is defined as

$$f_{\text{fac}} = \sum_{v \in V} \max_{u \in S} w_{u,v}. \tag{15}$$

Define $\hat{w}$ as k-NNG counterpart of w, i.e., $\hat{w}_{i,j} = w_{i,j}$ if j is among the k nearest neighbor of i, and $\hat{w}_{i,j} = 0$, otherwise. To establish that $\hat{f}^{\text{k-NNG}}_{\text{fac}}$, even with very sparse k, is a good approximation of f_{fac}, we rely on a key observation: $\max_{j \in S} w_{i,j} = \max_{j \in S} \hat{w}_{i,j}$ holds if the set S contains at least one item that is among the k nearest neighbor of i. Thus, showing that $\hat{f}^{\text{k-NNG}}_{\text{fac}}(S) = f_{\text{fac}}(S)$ is equivalent as showing that the set S contains at least one item that is among the k nearest neighbor of item i for any $i \in V$. Let's denote $\vec{w}_i = \{w_{i,1}, \ldots, w_{i,n}\}$ as the vector containing the weights on all edges out of the vertex i. To this end, we assume that the ranking of any item $j \in V$ among the vector $\vec{w}_i$ for any $i \in V$ is uniformly distributed over $\{1, 2, \ldots, n\}$ and that the ranking of j in one weight vector $\vec{w}_i$ is independent of its ranking in another.

Lemma 5. *For the facility location function, we have:*

$$\hat{f}^{\textit{k-NNG}}_{\textit{fac}}(S) = f_{\textit{fac}}(S), \forall S \subseteq V \textit{ s.t. } |S| \geq m, \tag{16}$$

with probability at least $(1 - \theta)$*, and the sparsity of the k-NNG being at least*

$$k = n[1 - (\frac{\theta}{n})^{\frac{1}{m}}]. \tag{17}$$

Assuming that m, n are in the same order and that θ is a constant, we have $\lim_{n \to \infty} n[1 - (\frac{\theta}{n})^{\frac{1}{m}}] = O(\log n)$. The Lemma implies that with high probability, $\hat{f}^{\text{k-NNG}}_{\text{fac}}$ and f_{fac} share the same function value for any sets of size greater than some threshold m, where the k-NNG can be as sparse as $k = O(\log n)$. Note that the result does not extend to general graph-based submodular functions. However, the class of facility location functions, for various applications, suffice to provide superior performance (Wei et al., 2013; Lin and Bilmes, 2009; Liu et al., 2013).

By Lemma 5, $\hat{f}^{\text{k-NNG}}_{\text{fac}}$ alone provides a good approximation for f_{fac}. It thus suffices, in this case, to apply a single-stage greedy algorithm (MULTGREED with $J = 1$) using $\hat{f}^{\text{k-NNG}}_{\text{fac}}$ as the surrogate. As a concrete example, consider an instance of the procedure with $\theta = 0.05$, $n = 10^6$, $k = 0.009n$, and $\ell = 0.1n$. Then, Lemma 5 implies that with probability 95%, $\hat{f}^{\text{k-NNG}}_{\text{fac}}(S) = f_{\text{fac}}(S)$ holds for any $|S| \geq 0.00186n$, giving an individual greedy ratio of $\alpha_i = 1$ for $0.00186n \leq i \leq 0.1n$. The greedy ratio α, defined as the harmonic mean of $\{\alpha_i\}_{i=1}^{\ell}$, is then bounded as $\alpha \leq 1.02$, which guarantees a solution in this instance close to optimum, thanks to Theorem 1.

Saturated coverage function: Successfully applied in document summarization (Lin and Bilmes, 2011), the saturated coverage function is another subclass of graph-based submodular functions, defined as

$$f_{\text{sat}}(S) = \sum_{v \in V} \min\{\sum_{u \in S} w_{v,u}, \xi \sum_{u \in V} w_{v,u}\}, \tag{18}$$

where $0 < \xi \leq 1$ is a hyperparameter that determines the saturation ratio. The class of uniform submodular mixtures includes f_{sat}. In this case, we can construct a two-stage greedy algorithm, where the modular upper bound $\hat{f}^{\text{mod}}$ and a sampling based function $\hat{f}^{\text{sub}}$ (with sampling probability p) are used as the two surrogates.

Lemma 6. *Given the saturated coverage function, an instance of* MULTGREED *with the size constraints* $\ell_1 = \lfloor \frac{n\xi}{(1-\xi)\gamma+\xi} \rfloor$ *and* $\ell_2 = \max\{0, \ell - \ell_1\}$ *(where* $\gamma = \frac{\max_{u,v} w_{u,v}}{\min_{(u,v)\in E(G)} w_{u,v}}$*, assuming all extent graph edges are positively weighted) yields a solution with the individual greedy ratios*

$$\alpha_i = 1, \textit{ for } i = 1, \ldots, \ell_1$$

And with probability $1 - \delta$*,*

$$1 \leq \alpha_i \leq \frac{1}{1-\epsilon}, \textit{ for } i = \ell_1 + 1 \ldots, \ell$$

where $\delta = (1 - 5n^{\ell} e^{-\frac{np(g^{\ell})^2\epsilon^2}{64B^2}})$ *and* $g^{\ell} = \max_{u \in V \setminus S_{\ell-1}} f(u|S_{\ell-1}) > 0$.

A main intuition of this result, is that f_{sat} is modular up to set of size ℓ_1. Hence it makes sense to use $\hat{f}^{\text{mod}}$ for these cases. Similarly for the second stage, it is reasonable to use $\hat{f}^{\text{sub}}$ with an appropriate p.

Feature based submodular function: Successfully applied in the speech data subset selection problem (Wei et al., 2014a;b), the feature based submodular function has the following form:

$$f_{\text{fea}} = \sum_{u \in \mathcal{F}} g(c_u(S)), \tag{19}$$

where g is concave in the form $g(x) = x^a$ for $0 < a \leq 1$, $\mathcal{F}$ is a set of "features", and $c_u(S) = \sum_{v \in S} c_u(s)$ is a non-negative modular score for feature $u \in \mathcal{F}$ in set S, with $c_u(v)$ measuring the degree to which item u possesses feature u. Maximizing this objective naturally encourages diversity and coverage of the features within the chosen set of items. Again, f_{fea} is a member of the class of uniform submodular mixtures. The curvature of f_{fea} is governed by the curvature of the concave function g and thus is determined by a. We can construct a two-stage procedure similar to that for f_{sat}, where we optimize over $\hat{f}^{\text{mod}}$ and $\hat{f}^{\text{sub}}$ with a suitable choice of the sampling probability p.

Lemma 7. *Given the feature based submodular function, an instance of* MULTGREED *with the size constraints being* ℓ_1 *and* ℓ_2*, yields a solution with the individual greedy ratio bounded as:*

$$1 \leq \alpha_i \leq O(i^{1-a}), \textit{ for } i = 1, \ldots, \ell_1$$

And with probability $1 - \delta$*,*

$$1 \leq \alpha_i \leq \frac{1}{1-\epsilon}, \textit{ for } i = \ell_1 + 1 \ldots, \ell$$

where $\delta = (1 - 5n^{\ell} e^{-\frac{np(g^{\ell})^2\epsilon^2}{64B^2}})$ *and* $g^{\ell} = \max_{u \in V \setminus S_{\ell-1}} f(u|S_{\ell-1}) > 0$.

The lemma implies that with an appropriate choice of the sampling probability p for $\hat{f}^{\text{sub}}$, the performance loss in the second stage could be negligible. However, there is some performance loss introduced by the first stage, depending on a and ℓ_1. The choices for ℓ_1 and ℓ_2 determine the tradeoff between loss of the performance guarantee and the computational reduction: larger ℓ_1 is chosen when computation is critical or when g is less curved (larger values of a), while larger ℓ_2 is chosen when algorithmic performance is the priority or g is more curved (smaller values of a).

Set cover function: Another important class of submodular functions is the set cover function. Given a set of sets $\{A_1, \ldots, A_{|V|}\}$, the universe $U = \cup_{v \in V} A_v$, and the weights $w : U \to \mathbb{R}$, the set cover function is defined as:

$$f_{\text{sc}}(S) = w(\cup_{v \in S} A_v), \tag{20}$$

where $w(A) = \sum_{u \in A} w(u)$ for $A \subseteq U$ with $w(u)$ being the weight of item $u \in U$. f_{sc} is again a uniform submodular mixture since it can be equivalently written as

$$f_{\text{sc}}(S) = \sum_{u \in U} \min\{c_u(S), 1\} w(u), \tag{21}$$

where $c_u(S)$ denotes the number of times that item $u \in U$ is covered by the set of sets $\{A_v : v \in S\}$. Thanks to Lemma 3, a single-stage procedure where we optimize over the sampling based surrogate $\hat{f}^{\text{sub}}$ with appropriate sampling probability p, suffices to provide a good performance guarantee along with a computational reduction.

Lemma 8. *Given a set cover function* f_{SC}*, such an instance of* MULTGREED *with the size constraints being* $\ell_1 = \ell$ *yields a solution with the individual greedy ratio bounded as the following:*

With probability $1 - \delta$*,*

$$1 \leq \alpha_i \leq \frac{1}{1-\epsilon}, \textit{ for } i = 1 \ldots, \ell$$

where $\delta = (1 - 5n^{\ell} e^{-\frac{np(g^{\ell})^2\epsilon^2}{64B^2}})$ *and* $g^{\ell} = \max_{u \in V \setminus S_{\ell-1}} f(u|S_{\ell-1}) > 0$.

6 Experiments

We empirically test the performance of MULTGREED for three of the submodular functions considered above. We address the following questions: 1) how well does MULTGREED perform compared to LAZYGREED, 2) how much relative time reduction can be achieved, 3) how well does the greedy ratio perform as a quality measure, 4) how well does the framework scale to massive data sets. We run experiments on two scenarios: 1) simulations with medium sized data, 2) real world speech data selection on millions of ground elements.

Simulations: All simulations are performed on the same data with size $|V| = 20,000$, formed by randomly sampling

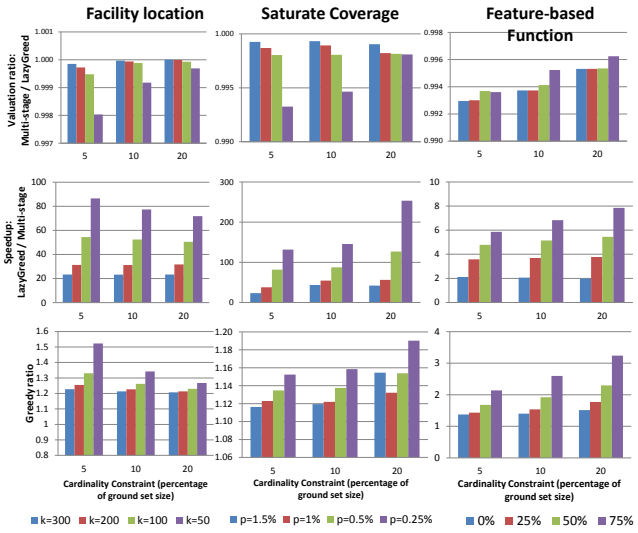

Figure 1. A comparison of the function values (top row), the running time (middle row), the greedy ratio (bottom row) between lazy greedy and our multi-stage approach, under different choices of the surrogate function for f_{fac} (left column), f_{sat} (middle column), and f_{fea} (right column).

	5%	10%	20%	all
Averaged Random	38.2	35.1	34.4	
Histogram-Entropy	37.6	34.2	fail	31.0
Multi-stage Submodular	**37.3**	**34.1**	**32.7**	

Table 1. Word error rates under averaged random, histogram-entropy, and the multi-stage submodular chosen subsets at various sized percentages (lower the better). Histogram-entropy result for the 20% condition is not available due to its objective's saturation after 10%.

from a large speech recognition training corpus (the "Fisher" corpus). Each sample pair has a similarity score, and the graph-based submodular functions f_{fac} and f_{sat} are instantiated using the corresponding similarity matrix. A set of features $\mathcal{F}$ sized $|\mathcal{F}| \approx 75000$ is derived from the same data to instantiate f_{fea}. In all runs of MULTGREED, we set $\{\beta_i\}_{i=1}^{\ell}$ using the schedule $\beta_i = c + \frac{(i-1)(1-c)}{\ell}$ with $c = 0.5$. Performance of MULTGREED and LAZYGREED is measured by the function valuations and the wall-clock running time. The optional stage of pruning is performed only for f_{fea}, but not for f_{fac} f_{sat}, since both f_{fac} and f_{sat} are very curved functions, for which the pruning procedure cannot effectively remove items.

For f_{fac}, use one stage with surrogates $\hat{f}^{\text{k-NNG}}$ with $k \in \{50, 100, 200, 300\}$. MULTGREED gives about a 20-80 times speedup over LAZYGREED with at least 99.8% of the standard greedy solution (first column of Fig. 1). For f_{sat}, the saturation ratio ξ is set as 0.25. Two stages using surrogate functions $\hat{f}^{\text{mod}}$ and $\hat{f}^{\text{sub}}$ are applied, under size constraints $\ell_1 = \lfloor \frac{n\xi}{5(1-\xi)+\xi} \rfloor = 0.05n$, and $\ell_2 = \ell - \ell_1$. We test $\hat{f}^{\text{sub}}$ with various sampling probabilities: $p \in \{0.25\%, 0.5\%, 1\%, 1.5\%\}$. The results (2nd column of Fig. 1) show a speedup of up to 250 with at least 99.25% the quality of LAZYGREED. Next, for f_{fea}, we set g to be the square root function. Two stages of surrogates $\hat{f}^{\text{mod}}$ and $\hat{f}^{\text{sub}}$ are applied. $\hat{f}^{\text{sub}}$ is defined on a randomly selected feature subset of size 37,500. We test with different combinations of size constraints ℓ_1 and ℓ_2 by setting $\ell_1 \in \{0, 0.25\ell, 0.5\ell, 0.75\ell\}$ with $\ell_2 = \ell - \ell_1$. This gives about a 2-8 times speedup with at least 99.3% of LAZYGREED quality (right column of Fig 1). Empirically, the greedy ratio is very tight since it is always close to 1. For most cases, it is a good indicator of the performance for function valuations, since lower values of α always lead to higher performance of MULTGREED. For f_{fac} and f_{sat}, the speedup reported does not include the potentially significant additional complexity reduction on graph-construction. Especially for f_{fac}, efficient algorithms exist for fast (approximate) construction of the k-NNG (Beygelzimer et al., 2006).

Speech Data Subset Selection: We next test the performance of MULTGREED on a very large-scale problem, where running even LAZYGREED is infeasible. We address the problem of speech data subset selection (King et al., 2005; Lin and Bilmes, 2009; Wei et al., 2013): given a massive (speech) data set for training automatic speech recognition (ASR) systems, we wish to select a representative subset that fits a given budget (measured in total time) and train a system only on the subset. The intended benefit is to significantly shorten ASR system development time, which is then only done on the representative subset of the data. Problem 5 addresses this where the objective is the facility location function f_{fac}, and the pair-wise similarity between speech samples is computed by kernels (Wei et al., 2013). We subselect 1300 hours of conversational English telephone data from the "Switchboard", "Switchboard Cellular", and "Fisher" corpora, which, in total, comprises 1,322,108 segments of speech (i.e., $|V| = n = 1,322,018$). The estimated running time of LAZYGREED with f_{fac} on such large data is at least a week. Rendering the full $O(n^2)$ similarity matrix is even more impractical due to memory requirements. We here test MULTGREED using $\hat{f}_{\text{fac}}^{\text{k-NNG}}$ with the sparsity of k-NNG set as $k = 1,000$. MULTGREED, then, runs in only a few minutes, yielding a speedup of more than a thousand over LAZYGREED! We measure the performance of the selection by the word error rate (WER) of the ASR system trained on the corresponding selected subset of the data. We test on different budget constraints (5%, 10% and 20% of the whole speech data). We compare our selection against two baseline selection methods: (1) averaged random method, where we randomly sample the data set at appropriate sizes, train different ASR systems for each set, and average their WER; (2) a non-submodular "histogram-entropy" based method, described in (Wu et al., 2007). Table 1 illustrates that our framework yields consistently superior results to these baselines.

7 Discussion

Certain other domains may be applicable to the analysis introduced in this paper. In the case of feature selection, for example, one may wish to optimize the mutual information function $f_{\text{mi}} = I(X_S; C)$ which either is not submodular, or can become submodular by assuming that the random variables X_V are independent given C (Krause and Guestrin, 2005a). In either case, however, the complexity of evaluating f_{mi} can be daunting, leading to previous work suggesting a tractable surrogate $\hat{f}_{\text{mi}}(S) = \sum_{v \in S} I(X_v; C) - \lambda \sum_{v,u \in S} I(X_v; X_u)$, where λ is a hyperparameter (Peng et al., 2005). Under certain assumptions, this surrogate is in fact equivalent to the original (Balagani and Phoha, 2010). Unnoticed by these authors, however, this function is submodular and non-monotone. We plan in the future to extend our framework to additionally handle such functions.

We would also like to extend these ideas to other submodular optimization scenarios, like submodular minimization (Fujishige and Isotani, 2011; Iyer et al., 2013a;b), and the class of optimization problems involving submodular constraints (Iyer and Bilmes, 2013) (which includes the submodular set cover, the submodular cost submodular set cover, and the submodular cost submodular knapsack). While many of the algorithms for these problems use proxy functions as surrogates, they often choose generic functions as proxies to obtain theoretical guarantees. It will be interesting to see if an intelligent design of surrogate functions, could yield better theoretical guarantees for real world problems.

Acknowledgments: We thank Shengjie Wang and Wenruo Bai for discussions. This work is partially supported by the Intelligence Advanced Research Projects Activity (IARPA) under agreement number FA8650-12-2-7263, the National Science Foundation under Grant No. IIS-1162606, and by a Google, a Microsoft, and an Intel research award.

References

A. Badanidiyuru and J. Vondrák. Fast algorithms for maximizing submodular functions. In *SODA*, 2014.

K. S. Balagani and V. V. Phoha. On the feature selection criterion based on an approximation of multidimensional mutual information. *PAMI, IEEE Transactions*, 2010.

A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbor. In *ICML*, 2006.

O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *CVPR*, 2008.

F. Chierichetti, R. Kumar, and A. Tomkins. Max-cover in Map-Reduce. In *WWW*, 2010.

M. Conforti and G. Cornuejols. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the Rado-Edmonds theorem. *Discrete Applied Mathematics*, 1984.

U. Feige. A threshold of ln n for approximating set cover. *JACM*, 1998.

S. Fujishige. *Submodular functions and optimization*, volume 58. Elsevier Science, 2005.

S. Fujishige and S. Isotani. A submodular function minimization algorithm based on the minimum-norm base. *Pacific Journal of Optimization*, 7:3–17, 2011.

R. Iyer and J. Bilmes. Algorithms for approximate minimization of the difference between submodular functions, with applications. *In UAI*, 2012.

R. Iyer and J. Bilmes. Submodular optimization with submodular cover and submodular knapsack constraints. In *NIPS*, 2013.

R. Iyer, S. Jegelka, and J. Bilmes. Curvature and Optimal Algorithms for Learning and Minimizing Submodular Functions. *NIPS*, 2013a.

R. Iyer, S. Jegelka, and J. Bilmes. Fast semidifferential based submodular function optimization. In *ICML*, 2013b.

D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

S. King, C. Bartels, and J. Bilmes. SVitchboard 1: Small vocabulary tasks from switchboard 1. In *European Conf. on Speech Communication and Technology (Eurospeech)*, Lisbon, Portugal, September 2005.

V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE TPAMI*, 26(2):147–159, 2004.

A. Krause and C. Guestrin. Near-optimal nonmyopic value of information in graphical models. In *UAI*, 2005a.

A. Krause and C. Guestrin. A note on the budgeted maximization of submodular functions, 2005b. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.130.3314.

A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *JMLR*, 9:235–284, 2008.

A. Kulesza and B. Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2-3):123–286, 2012.

R. Kumar, B. Moseley, S. Vassilvitskii, and A. Vattani. Fast greedy algorithms in mapreduce and streaming. In *SPAA*, 2013.

J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *SIGKDD*, 2007.

H. Lin and J. Bilmes. How to select a good training-data subset for transcription: Submodular active selection for sequences. In *Interspeech*, 2009.

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

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

Y. Liu, K. Wei, K. Kirchhoff, Y. Song, and J. Bilmes. Submodular feature selection for high-dimensional acoustic score spaces. In *ICASSP*, 2013.

M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. *Optimization Techniques*, 1978.

B. Mirzasoleiman, A. Karbasi, R. Sarkar, and A. Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *NIPS*, 2013.

G. Nemhauser and L. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of Operations Research*, 3(3):177–188, 1978.

G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, (1), 1978.

H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *PAMI, IEEE Transactions*, 2005.

R. Shah, R. Iyer, and S. Chaudhuri. Object mining for large video data. *Proc. BMVC*, 22(10):761–767, 2011.

Y. Singer. How to win friends and influence people, truthfully: influence maximization mechanisms for social networks. In *WSDM*. ACM, 2012.

P. Stobbe and A. Krause. Efficient minimization of decomposable submodular functions. In *NIPS*, 2010.

M. Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1):41–43, 2004.

K. Wei, Y. Liu, K. Kirchhoff, and J. Bilmes. Using document summarization techniques for speech data subset selection. In *NAACL/HLT*, 2013.

K. Wei, Y. Liu, K. Kirchhoff, C. Bartels, and J. Bilmes. Submodular subset selection for large-scale speech training data. In *ICASSP*, 2014a.

K. Wei, Y. Liu, K. Kirchhoff, and J. Bilmes. Unsupervised submodular subset selection for speech data. In *ICASSP*, 2014b.

L. A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.

Y. Wu, R. Zhang, and A. Rudnicky. Data selection for speech recognition. In *ASRU*, 2007.

Appendix

Proof of Lemma 1

Proof. Let $s_1, \ldots, s_\ell$ be the sequence of items selected by an instance of the LAZYGREED. We denote that $S_i = \{s_1, \ldots, s_i\}$, such that $S_1 \subset S_2 \subset \cdots \subset S_\ell$. Consider the sequence of items $\{u_1, \ldots, u_n\}$ ordered non-increasingly in terms of their gain conditioned on all other items, i.e., $f(u_1|V \setminus u_1) \geq \cdots \geq f(u_n|V \setminus u_n)$. Without loss of generality, we assume that $S_\ell \neq \{u_1, \ldots, u_\ell\}$, otherwise, the Lemma follows trivially. Next, we show the following:

$$f(s_i|S_{i-1}) \geq f(u_i|V \setminus u_i), \forall i = 1, \ldots, \ell. \tag{22}$$

We consider two cases:

(1) $u_i \notin S_{i-1}$

Consider the following:

$$\begin{aligned} f(s_i|S_{i-1}) &= \max_{v \in V} f(v|S_{i-1}) \\ &\geq f(u_i|S_{i-1}) \\ &\geq f(u_i|V \setminus u_i) \end{aligned}$$

(2) $u_i \in S_{i-1}$

Since S_{i-1} is of size $i-1$ and contains an item u_i, S_{i-1} cannot include all items in the set $\{u_1, \ldots, u_{i-1}\}$. Therefore, there exists an index $t \leq i-1$ such that $u_t \notin S_{i-1}$. We have the following:

$$\begin{aligned} f(s_i|S_{i-1}) &= \max_{v \in V} f(v|S_{i-1}) \\ &\geq f(u_t|S_{i-1}) \\ &\geq f(u_t|V \setminus u_t) \\ &\geq f(u_i|V \setminus u_i) \end{aligned}$$

For items $j \in V \setminus \hat{V}$, the singleton gain can be bounded as the following:

$$f(j) < f(u_\ell|V \setminus u_\ell) \leq f(s_\ell|S_{\ell-1})$$

The singleton gain of the items in the removed data set is strictly less than smallest the marginal gain of the selected item by the greedy algorithm, hence, it is guaranteed that the greedy selection does not pick items from $V \setminus \hat{V}$, hence the pruning procedure does not affect the performance of the standard greedy procedure. □

Proof of Theorem 1

Proof. We employ similar proof techniques from (Conforti and Cornuejols, 1984), where they show the curvature dependent bound for Problem 1. We generalize their results by introducing the greedy ratio in the bound.

To prove the theorem, we need the following two Lemmas.

Lemma 9.

$$\prod_{i=1}^{n}(1 - \frac{1}{x_i}) \leq (1 - \frac{1}{\bar{x}_h})^n \tag{23}$$

where $x_i \geq 1, \forall i = 1, \ldots, n$, and $\bar{x}_h$ is harmonic mean of x_i's, i.e., $\bar{x}_h = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$.

Proof. Consider the fact that geometric mean of any non-negative vector lower bounds arithmetic mean of the same vector, namely,

$$(\prod_{i=1}^{n} z_i)^{1/n} \leq \frac{\sum_{i=1}^{n} z_i}{n} \tag{24}$$

where $z_i \geq 0, \forall i$.
Let $z_i = 1 - \frac{1}{x_i}$, where $x_i \geq 1$, we have

$$(\prod_{i=1}^{n}(1-\frac{1}{x_i}))^{1/n} \leq \frac{\sum_{i=1}^{n}(1-\frac{1}{x_i})}{n} \tag{25}$$

Then,

$$\prod_{i=1}^{n}(1-\frac{1}{x_i}) \leq (\frac{\sum_{i=1}^{n}(1-\frac{1}{x_i})}{n})^n \tag{26}$$

Rearrange the term in the R.H.S,

$$\frac{\sum_{i=1}^{n}(1-\frac{1}{x_i})}{n} = 1 - \frac{\sum_{i=1}^{n}\frac{1}{x_i}}{n} \tag{27}$$

$$= 1 - \frac{1}{\frac{n}{\sum_{i=1}^{n}\frac{1}{x_i}}} \tag{28}$$

$$= 1 - \frac{1}{\bar{x}_h} \tag{29}$$

□

Lemma 10. *For $t = 0, \ldots, \ell-1$,*

$$\sum_{i:s_i \in S^t \cap S^{OPT}} f(s_i|S^{i-1}) + \kappa_f \sum_{i:s_i \in S^t \setminus S^{OPT}} f(s_i|S^{i-1}) + \alpha_{t+1}(\ell - |S^t \cap S^{OPT}|) f(s_{t+1}|S^t) \geq f(S^{OPT}),$$

where $S^i = \{s_1, \ldots, s_i\}$, ℓ is the size constraint, κ_f is the total curvature of f, and α_i is the individual greedy ratio for iteration i.

Proof. By submodularity and definition of the greedy ratio, we have the following inequality:

$$f(S^{\text{OPT}} \cup S^t) \leq f(S^t) + \sum_{u \in S^{\text{OPT}} \setminus S^t} f(u|S^t) \tag{30}$$

$$\leq f(S^t) + \sum_{u \in S^{\text{OPT}} \setminus S^t} \alpha_{t+1} f(s_{t+1}|S^t) \tag{31}$$

$$= f(S^t) + |S^{\text{OPT}} \setminus S^t| \alpha_{t+1} f(s_{t+1}|S^t) \tag{32}$$

By the definition of curvature, we have

$$f(S^{\text{OPT}} \cup S^t) \tag{33}$$

$$= f(S^{\text{OPT}}) + \sum_{i:s_i \in S^t \setminus S^{\text{OPT}}} f(s_i|S^{\text{OPT}} \cup S^{i-1}) \tag{34}$$

$$\geq f(S^{\text{OPT}}) + (1-\kappa_f) \sum_{i:s_i \in S^t \setminus S^{\text{OPT}}} f(s_i|S^{i-1}) \tag{35}$$

Combining inequalities 32 and 35, we can get the following bound:

$$f(S^{\text{OPT}}) + (1-\kappa_f) \sum_{i:s_i \in S^t \setminus S^{\text{OPT}}} f(s_i|S^{i-1}) \leq f(S^t) + |S^{\text{OPT}} \setminus S^t| \alpha_{t+1} f(s_{t+1}|S^t)$$

Then,

$$f(S^{\text{OPT}}) + (1-\kappa_f) \sum_{i:s_i \in S^t \setminus S^{\text{OPT}}} f(s_i|S^{i-1}) \leq \sum_{s_i \in S^t} f(s_i|S^{i-1}) + (\ell - |S^t \cap S^{\text{OPT}}|)\alpha_{t+1} f(s_{t+1}|S^t)$$

□

Consider the family $\mathcal{F}_{f,\ell,\kappa}$ of all instances of Problem 1. For notation simplicity we write $\mathcal{F} = \mathcal{F}_{\{,\ell,\kappa}$. Given a problem instance, we assume that the multi-stage greedy algorithm MULTGREED returns a subset S^ℓ and $|S^\ell \cap S^{\text{OPT}}| = m$. For $0 \leq m \leq \ell$, let $1 \leq i_1 < i_2 <, \ldots, < i_m \leq \ell$ be a sequence of integers such that $S^\ell = \{s_1, s_2, \ldots, s_\ell\}$ has the elements $s_{i_1}, \ldots, s_{i_m}$ in common with the optimal solution S^{OPT}. Note that when $m = 0$, the set of common elements is empty. By Lemma 10, it's easy to verify that $\frac{f(S^\ell)}{f(S^{\text{OPT}})} \geq B(i_1, \ldots, i_m)$, where $B(i_1, \ldots, i_m)$ is the solution to the following optimization problem:

$$\min_{\rho_1,\ldots,\rho_\ell} \quad \mathbb{1}^T \rho \tag{36}$$

$$\text{subject to} \quad \rho_i \geq 0, i = 1, \ldots \ell \tag{37}$$

$$A(i_1, \ldots, i_m)\rho \geq 1 \tag{38}$$

Note the matrix $A(i_1, \ldots, i_m)$ is defined the same as the matrix used in the proof of Lemma 5.1 in (Conforti and Cornuejols, 1984), except that we have all the diagonal entries of the matrix scaled by α_i for each i. Since the matrix $A(i_1, \ldots, i_m)$ is full rank, the polyhedron given by $A(i_1, \ldots, i_m)\rho \geq 1$ has only one vertex, and solving $A(i_1, \ldots, i_m)\rho = 1$ yields the vertex $\rho^* \geq 0$. Note that the optimum of an LP can always be obtained at a vertex,the only feasible vertex ρ^* is guaranteed to be among the optimum. Therefore ρ^* is also the solution to the linear program. Hence, $B(i_1, \ldots, i_m) = \sum_{i=1}^{\ell} \rho_i^*$.

Next,we are going to show that $B(i_1, \ldots, i_m) \geq B(\emptyset)$. We start with the case where $m = 1$, and assume that $S^\ell \cap S^{\text{OPT}} = \{s_p\}, 1 \leq p \leq \ell$. We denote the solution to the problem $B(p)$ as ρ', and the solution to $B(\emptyset)$ as ρ. Solving for ρ and ρ' yields the following:

$$\rho_i = \frac{1}{\alpha_i \ell} \prod_{j=1}^{i-1}(1 - \frac{\kappa}{\alpha_j \ell}), \forall i = 1, \ldots, \ell$$

$$\rho'_i = \rho_i, \forall i = 1, \ldots, p;$$

$$\rho'_p = \frac{1}{(\ell-1)\alpha_p}(1 - \frac{1}{\ell\alpha_p}) \prod_{j=1}^{p-1}(1 - \frac{\kappa}{\ell\alpha_j})$$

$$\rho'_i = \frac{1}{(\ell-1)\alpha_i}(1 - \frac{1}{\ell\alpha_p}) \prod_{j=1}^{p-1}(1 - \frac{\kappa}{\ell\alpha_j}) \prod_{j=p=1}^{i-1}(1 - \frac{\kappa}{(\ell-1)\alpha_j}),$$

for $i = p+1, \ldots, \ell$;

To show that $B(p) \geq B(\emptyset)$, it suffices to show that $\rho_i \leq \rho'_i$ for any i, equivalently $\frac{\rho_i}{\rho'_i} \leq 1$. Consider the following for $i = p, \ldots, \ell$

$$\frac{\rho_i}{\rho'_i} = \frac{\ell-1}{\ell} \frac{1-\frac{\kappa}{\ell\alpha_p}}{1-\frac{1}{\ell\alpha_p}} \prod_{j=p+1}^{i-1} \frac{1-\frac{\kappa}{\ell\alpha_j}}{1-\frac{\kappa}{(\ell-1)\alpha_j}} \tag{39}$$

$$\leq \frac{\ell-1}{\ell} \frac{1-\frac{\kappa}{\ell}}{1-\frac{1}{\ell}} \prod_{j=p+1}^{i-1} \frac{1-\frac{\kappa}{\ell}}{1-\frac{\kappa}{\ell-1}} \tag{40}$$

$$\leq \frac{(1-\frac{\kappa}{\ell})^\ell - 1}{(1-\frac{\kappa}{(\ell-1)})^\ell - 2} \tag{41}$$

$$\leq 1 \tag{42}$$

The first inequality follows since $\frac{1-\frac{\kappa}{\ell\alpha}}{1-\frac{\kappa}{(\ell-1)\alpha}}$ is monotonically non-increasing with α, hence, its maximum is attained when $\alpha = 1$. The last inequality follows from the fact that $(1 - \frac{\kappa}{x})^{x-1}$, where $\kappa \in [0, 1]$, is monotonically non-increasing when $x \geq 1$. We have shown that $\rho_i \leq \rho'_i$ for any i, hence, $B(\emptyset) \leq B(p)$. We are going to show that $B(i_1, \ldots, i_m) \geq B(\emptyset)$ by induction. It's already proved that it holds when $m = 1$. Suppose $B(i_1, \ldots, i_q) \geq B(\emptyset)$ holds, we wish to show the following: $B(i_1, \ldots, i_q, i_{q+1}) \geq B(i_1, \ldots, i_q)$ for any $i_{q+1} \notin \{i_1, \ldots, i_q\}$. We denote the solution to problem $B(i_1, \ldots, i_q)$ as ρ^{B_q}, and the solution to problem $B(i_1, \ldots, i_{q+1})$ as $\rho^{B_{q+1}}$. It's easy to see that $\rho_i^{B_{q+1}} = \rho_i^{B_q}$, for $i = 1, \ldots, i_{q+1}$. Let $r = i_{q+1}$, then for $i = r+1, \ldots, \ell$, we have the following:

$$\frac{\rho_i^{B_q}}{\rho_i^{B_{q+1}}} = \frac{\frac{1}{(\ell-q)\alpha_i}}{\frac{1}{(\ell-1-1)\alpha_i}} \frac{(1-\frac{\kappa}{(\ell-q)\alpha_r})}{1-\frac{1}{(\ell-q)\alpha_r}} \prod_{j=r+1}^{i-1} \frac{1-\frac{\kappa}{(\ell-q)\alpha_j}}{1-\frac{\kappa}{(\ell-q-1)\alpha_j}} \tag{43}$$

$$\leq \frac{\ell-q-1}{\ell-q} \frac{1-\frac{\kappa}{\ell-q}}{1-\frac{1}{\ell-q}} (\frac{1-\frac{\kappa}{\ell-q}}{1-\frac{\kappa}{\ell-q-1}})^{i-1-r} \tag{44}$$

$$\leq \frac{(1-\frac{\kappa}{\ell-q})^{\ell-1-1}}{(1-\frac{\kappa}{\ell-q-1})^{\ell-q-2}} \tag{45}$$

$$\leq 1 \tag{46}$$

Therefore, we have $\rho_i^{B_q} \leq \rho_i^{B_{q+1}}$ for all i, thus $B(i_1, \ldots, i_q) \leq B(i_1, \ldots, i_{q+1})$.

Then, we have the following:

$$\begin{aligned}
\frac{f(S^\ell)}{f(S^{\text{OPT}})} &\geq B(i_1, \ldots, i_m) \\
&\geq B(\emptyset) \\
&= \frac{1}{\kappa}(1 - \prod_{i=1}^{\ell}(1 - \frac{\kappa}{\ell\alpha_i})) \\
&\geq \frac{1}{\kappa}(1 - e^{-\frac{\kappa}{\alpha}})
\end{aligned}$$

where α is the harmonic mean of α_i's.

To show that the approximation factor is tight. We only show for the case where $\kappa = 1$. Results could be generalized for any value of κ.

We use the similar tight instance construction as in (Nemhauser and Wolsey, 1978), where they show a class of instances of the greedy algorithm that achieves the approximate factor $(1-(1-\frac{1}{\ell})^\ell)$. In this case, we are going to show an instance of MULTGREED that achieves the approximate factor $(1 - (1 - \frac{1}{\alpha}\ell)^\ell)$, where α is the greedy ratio of this instance of MULTGREED. To this end, let's define a family of submodular functions that provide worst-case examples over all possible $\alpha \geq 1$ and ℓ. The (α, ℓ)th subfamily is specified by function g_α^ℓ. Notice that g_α^ℓ is a set function on a ground set V of cardinality n. For notation simplicity, we drop the dependency on n in the set function g_α^ℓ, although there is a different function for each n. We will show how to construct a submodular function for a combination of n, α, ℓ, on which MULTGREED achieves the approximation factor $(1 - (1 - \frac{1}{\alpha\ell})^\ell)$.

Consider ground set $V = \{1, 2, \ldots, n\}$ that contains two types of elements *special* and *plain*. The subset M, with $|M| = \ell$, is the set of special elements and $V \setminus M$ is the set of plain elements. Value of $g_\alpha^\ell(S), S \subseteq V$, depends only on $|S|, |S \cap M|, \alpha$ and ℓ. For this reason, we write $g_\alpha^\ell(S) = g_\alpha^\ell(|S \cap M|, |S|) = g_\alpha^\ell(i, j)$, where $i = |S \cap M|$ and $j = |S|, i = 0, \ldots, \min(j, \ell), j = 0, \ldots, n$. Let $g_\alpha^\ell(i, j)$ for general ℓ, α, i, j be defined as following:

$$g_\alpha^\ell(0, j) = \ell(1 - (1 - \frac{1}{\alpha\ell})^j), 0 \leq j \leq n; \tag{47}$$

$$g_\alpha^\ell(i, j) = g_\alpha^\ell(0, j-i) + i\frac{\ell - g_\alpha^\ell(0, j-i)}{\ell}, \tag{48}$$

$$\text{for } 1 \leq i \leq \ell, i \leq j \leq n; \tag{49}$$

The fact that g_α^ℓ is monotone submodular is proved by tedious enumeration of the definition of submodularity and monotonicity case by case. We leave out the proof here and defer readers to (Nemhauser and Wolsey, 1978) for detail. Consider an instance of MULTGREED performed on g_α^ℓ, that only choose items from $V \setminus M$. Then, the greedy ratio would be:

$$\alpha_i = \max_{u \in V} \frac{f(u|S_{i-1})}{f(s_i|S_{i-1})} = \frac{g_\alpha^\ell(1, i) - g_\alpha^\ell(0, i-1)}{g_\alpha^\ell(0, i) - g_\alpha^\ell(0, i-1)} = \alpha \tag{50}$$

Hence, the harmonic mean of α_i's is equal to α. Output subset of MULTGREED is a subset of $V \setminus M$ with cardinality ℓ, thus has function value $g_\alpha^\ell(0, \ell) = \ell(1-(1-\frac{1}{\alpha\ell})^\ell)$, while optimal subset that maximizes submodular function f under cardinality constraint ℓ should be M and has function value $g_\alpha^\ell(M) = g_\alpha^\ell(\ell, \ell) = \ell$. Approximation factor becomes $\frac{g_\alpha^\ell(0,\ell)}{g_\alpha^\ell(M)} = 1 - (1 - \frac{1}{\alpha\ell})^\ell$. Therefore, the bound in terms of the greedy ratio α is tight. □

Proof of Theorem 2

Proof. Following the similar proof techniques from (Nemhauser et al., 1978) along with the definition of the knapsack greedy ratio in Eqn 6, we have the following:

$$f(S^{\text{OPT}}) \leq \frac{\alpha_i B}{c(s_i)} f(s_i|S_{i-1}) + f(S_{i-1}), \forall i = 1, \ldots, N \tag{51}$$

Rearrange the inequality, we get the following:

$$f(S^{\text{OPT}}) - f(S_i) \leq (1 - \frac{c(s_i)}{\alpha_i B})(f(S^{\text{OPT}}) - f(S_{i-1}))$$

holds for $i = 1, \ldots, N$. Then, we can obtain the following:

$$f(S_N) \geq (1 - \prod_{i=1}^{N}(1 - \frac{c(s_i)}{\alpha_i B}))f(S^{\text{OPT}})$$

By definition of the knapsack greedy ratio in Eqn 7 and Lemma 9, we obtain the following:

$$\begin{aligned} f(S_N) &\geq (1 - (1 - \frac{B'}{\alpha B N})^N) f(S^{\text{OPT}}) \\ &\geq (1 - e^{-\frac{B'}{B\alpha}}) f(S^{\text{OPT}}) \\ &\geq (1 - e^{-\frac{1}{\alpha}}) f(S^{\text{OPT}}) \end{aligned}$$

Since S_N is not a feasible solution to Problem 5, but S_{N-1} is. The knapsack greedy algorithm compares the solution between S_{N-1} and the maximum single value $f(v^*)$, where $v^* \in \text{argmax}_{u \in V; c(u) \leq B} f(u)$, and outputs the maximum of the two. Then, we can bound the output solution as follows:

$$\begin{aligned} \max\{f(v^*), f(S_{N-1})\} &\geq \frac{1}{2}(f(v^*) + f(S_{N-1})) \\ &\geq \frac{1}{2}(f(s_N) + f(S_{N-1})) \\ &\geq \frac{1}{2} f(S_N) \\ &\geq \frac{1}{2}(1 - e^{-\frac{1}{\alpha}}) f(S^{\text{OPT}}) \end{aligned}$$

□

Proof of Theorem3

This Theorem immediately follows from Theorem 2 and Theorem 3.1 of (Iyer and Bilmes, 2013).

Proof of Lemma 3

Proof. We employ methods that are similarly presented in (Mirzasoleiman et al., 2013), where they show the theoretical guarantee of applying $\hat{f}^{\text{sub}}$ as a surrogate function for the class of decomposable submodular functions. For notation simplicity, we write $\hat{f}^{\text{sub}}$ as $\hat{f}$. We denote the ground set size to be $|V| = n$. The expected size of the subset $\mathcal{T}' \subseteq \mathcal{T}$ is np. We are going to show that $\hat{f}$ is close to the target function f for all possible subset S of size $|S| \leq \ell$ with high probability, where ℓ is the size constraint of the selection. Given a fixed $S \subseteq V$, we can treat all $f_i(S)$'s as independent random variables and, by assumption, they are bounded as $0 \leq f_i(S) \leq \mathcal{B}$.

By Chernoff bound, we can bound the probability $\Pr(|\mathcal{T}'| \leq \frac{1}{2}np) \leq e^{-\frac{np}{2}}$. By the Hoeffding inequality, we can bound the probability $\Pr(|\hat{f}(S) - \mu| \geq \epsilon) \leq 2e^{-\frac{2n\epsilon^2}{\mathcal{B}^2}}$ and $\Pr(|\hat{f}(S) - \mu| \geq \epsilon) \leq 2e^{-\frac{np\epsilon^2}{\mathcal{B}^2}}$, where μ is the mean of the random variable $f_i(S)$ for any i.

By union bound, we have the following for ϵ being small such that $\frac{\epsilon^2}{\mathcal{B}^2} \leq \frac{1}{2}$.

$$\begin{aligned} \Pr(|f(S) - \hat{f}(S)| \leq 2\epsilon) &\geq 1 - 2e^{-\frac{2n\epsilon^2}{\mathcal{B}^2}} - 2e^{-\frac{np\epsilon^2}{\mathcal{B}^2}} - e^{-\frac{np}{2}} \\ &\geq 1 - 5e^{-\frac{np\epsilon^2}{\mathcal{B}^2}} \end{aligned}$$

There are in total n^ℓ sets of size less or equal to ℓ. By the union bound again, we can have the following:

$$\Pr(|f(S) - \hat{f}(S)| \leq 2\epsilon, \forall S \subseteq V, |S| \leq \ell) \geq 1 - 5n^\ell e^{-\frac{np\epsilon^2}{\mathcal{B}^2}} \tag{52}$$

Notice that this bound is meaningful only when $1 - 5n^\ell e^{-\frac{np\epsilon^2}{\mathcal{B}^2}} > 0$, in other words, we can obtain meaningful theoretical guarantee when the ground set size n is sufficiently large.

Now, we can analyze the performance guarantee in terms of the greedy ratio. Continuing from the results in the inequality 52 , we have that

$$|f(j|S \setminus j) - \hat{f}(j|S \setminus j)| \leq 4\epsilon$$

for any $j \in V$, and $|S| \leq \ell$, with probability $(1 - 5n^\ell e^{-\frac{np\epsilon^2}{\mathcal{B}^2}})$. Let the gain in the last iteration of the greedy algorithm to be g_G^ℓ, and assume that $g_G^\ell > 0$. Let t_i be the item that attains the maximum marginal gain by applying the target function f and s_i be the item selected by the surrogate function $\hat{f}$ for iteration i, then we have the following

$$\begin{aligned} \alpha_i &= \frac{f(t_i|S_{i-1})}{f(s_i|S_{i-1})} \leq \frac{f(t_i|S_{i-1})}{\hat{f}(s_i|S_{i-1}) - 4\epsilon} \leq \frac{f(t_i|S_{i-1})}{\hat{f}(t_i|S_{i-1}) - 4\epsilon} \\ &\leq \frac{f(t_i|S_{i-1})}{f(t_i|S_{i-1}) - 8\epsilon} = \frac{1}{1 - \frac{8\epsilon}{g_G^\ell}} = \frac{1}{1 - \epsilon'} \end{aligned}$$

where $\epsilon' = \frac{8\epsilon}{g_G^\ell}$. To formalize the result in terms of the greedy ratio, we can claim that with probability $(1 - 5n^\ell e^{-\frac{np(g_G^\ell)^2\epsilon'^2}{64\mathcal{B}^2}})$, we can bound the greedy ratio in each iteration as $\alpha_i \leq \frac{1}{1-\epsilon'}$. □

Proof of Lemma 4

Proof. Let t_i be the item with maximum marginal gain in iteration i, i.e., $t_i \in \text{argmax}_{u \in V} f(u|S_{i-1})$ By definition of the greedy ratio, we have the following:

$$\alpha_i = \frac{f(t_i|S_{i-1})}{f(s_i|S_{i-1})} \leq \frac{f(t_i)}{f(s_i|S_{i-1})} \leq \frac{f(s_i)}{f(s_i|S_{i-1})} \leq \frac{1}{1-\kappa_f(S_{i-1})}$$

The second inequality follows from that fact that s_i is the item greedily selected by the modular proxy $\hat{f}^{\text{mod}}$, therefore $f(s_i) = \max_{u \in V \setminus S_{i-1}} f(u) \geq f(t_i)$. The last inequality follows from the definition of the curvature. □

Proof of Lemma 5

Proof. Let $\vec{\hat{w}}_i$ be the ith row vector obtained from the k-NNG approximation from the full graph. Then, $\vec{\hat{w}}_i$ is the approximate vector for $\vec{w}_i$ with only k largest values retained. The key observation for the facility location function is that $f_{\text{fac}}(S) = \hat{f}^{\text{k-NNG}}_{\text{fac}}(S)$, if the set S contains items that are among the top k values of the row vector $\vec{w}_i$ for all i, since $\max_{j \in S} \vec{w}_i(j) = \max_{j \in S} \vec{\hat{w}}_i(j)$, if S contains items that are among the top k values of $\vec{w}_i$.

For notation simplicity, we write $\hat{f}$ for $\hat{f}^{\text{k-NNG}}_{\text{fac}}$ and f for f_{fac}. For any item $t \in V$, we have the probability of $w_{i,t}$ not being among the top k elements of the row vector w_i as $\frac{n-k}{n}$, given the uniform distribution assumption.

By the independence assumption, the probability, for which a set S_m of size m contain at least one item among the top k elements for each row vector, can be then computed as $[1-(\frac{n-k}{n})^m]^n$.

Let the probability that S_m covers among the top k elements of all row vectors be $1-\theta$. Then, we have the following:

$$[1-(\frac{n-k}{n})^m]^n = 1-\theta$$

Simplify the equation, we can get the following:

$$k = n[1-(1-(1-\theta)^{\frac{1}{n}})^{\frac{1}{m}}] \quad (53)$$

$$\approx n[1-(1-e^{-\frac{\theta}{n}})^{\frac{1}{m}}] \quad (54)$$

$$\approx n[1-(\frac{\theta}{n})^{\frac{1}{m}}] \quad (55)$$

The first approximation follows since $(1-\theta)^{\frac{1}{n}} \approx e^{-\frac{\theta}{n}}$, for θ being close to 0. The second approximation follows from that $e^{-\frac{\theta}{n}} \approx 1-\frac{\theta}{n}$, with $-\frac{\theta}{n} \approx 0$. □

Proof of Lemma 6

Proof. The bound of the individual greedy ratio α_i's for $i = \ell_1+1, \ldots, \ell$ can be immediately derived from Lemma 3. Therefore, it is left to show that $\alpha_i = 1$ for $i = 1, \ldots, \ell_1$, which, by Lemma 4, is equivalently as to show that $\kappa_f(S) = 0$ for $|S| \leq \ell_1$. For notation simplicity, we write f_{sat} as f. To show that $\kappa_f(S) = 0$, we can equivalently show that $f(S) = \sum_{j \in S} f(j)$ for any S of size $|S| \leq \lfloor \frac{n\xi}{(1-\xi)\gamma+\xi} \rfloor$.

Consider the following for any $S \subseteq V$ such that $|S| \leq \lfloor \frac{n\xi}{(1-\xi)\gamma+\xi} \rfloor$:

$$\frac{\sum_{j \in S} w_{i,j}}{\sum_{j \in V} w_{i,j}} = \frac{\sum_{j \in S} w_{i,j}}{\sum_{j \in S} w_{i,j} + \sum_{j \in V \setminus S} w_{i,j}}$$
$$\leq \frac{|S|\rho_{\max}}{|S|\rho_{\max} + (n-|S|)\rho_{\min}}$$
$$= \frac{|S|\gamma}{|S|\gamma + (n-|S|)}$$
$$\leq \frac{1}{1 + \frac{n}{\gamma \frac{n\xi}{(1-\xi)\gamma+\xi}} - \frac{1}{\frac{n\xi}{(1-\xi)\gamma+\xi}}}$$
$$= \xi$$

Without loss of generality, we can assume that all $w_{i,j} > 0$ for any $j \in S$ and a given i, since the above holds as well for $S' = \{j \in S | w_{i,j} > 0, \forall i\}$. Therefore, we have that $\sum_{j \in S} w_{i,j} \leq \xi \sum_{j \in V} w_{i,j}$ for all i. From which, we conclude that $\kappa_f(S) = 0$, for any $|S| \leq \lfloor \frac{n\xi}{(1-\xi)\gamma+\xi} \rfloor$ □

Proof of Lemma 7

Proof. The bound of the individual greedy ratio α_i's for $i = \ell_1+1, \ldots, \ell$ can be immediately derived from Lemma 3. Therefore, it is left to show that $\alpha_i \leq O(i^{1-a})$ for $i = 1, \ldots, \ell_1$. For notation simplicity, we write f_{fea} as f. Let $\rho_{\min} = \min_{u \in \mathcal{F}, v \in V : c_u(v) > 0} c_u(v)$ and $\rho_{\max} = \max_{u \in \mathcal{F}, v \in V} c_u(v)$. It suffices to show the following:

$$\alpha_i \leq \frac{1}{(1+(i-1)\gamma)^a - ((i-1)\gamma)^a},$$

where $\gamma = \frac{\rho_{\max}}{\rho_{\min}}$. It is easy to verify that

$$\frac{1}{(1+(i-1)\gamma)^a - ((i-1)\gamma)^a} = O(i^{1-a})$$

given γ is a constant. Let t_i denote the item with maximum marginal gain in iteration i, i.e., $t_i \in \text{argmax}_{u \in V \setminus S_{i-1}} f(u|S_{i-1})$, and s_i denote the item selected by using the modular proxy $\hat{f}^{\text{mod}}$. Consider the following:

$$\alpha_i = \frac{f(t_i|S_{i-1})}{f(s_i|S_{i-1})} \leq \frac{f(t_i)}{f(s_i|S_{i-1})} \leq \frac{f(s_i)}{f(s_i|S_{i-1})}$$
$$= \frac{\sum_{f \in \mathcal{F}} [c_f(s_i)]^a}{\sum_{f \in \mathcal{F}} [c_f(s_i) + c_f(S_{i-1})]^a - [c_f(S_{i-1})]^a}$$
$$\leq \frac{\sum_{f \in \mathcal{F}} [c_f(s_i)]^a}{\sum_{f \in \mathcal{F}} [c_f(s_i) + (i-1)\rho_{\max}]^a - [(i-1)\rho_{\max}]^a}$$

Let's consider the following function:

$$h(x_1, \ldots, x_n) = \frac{\sum_{i=1}^{n}(x_i)^a}{\sum_{i=1}^{n}((x_i + C)^a - C^a)} \tag{56}$$

where C is a constant. The function h is symmetric about all its variables $x_1, \ldots, x_n$. Notice that the function $h()$ is not convex or concave in its variables. However, we still want to maximize the function over the vector x within the range $[\rho_{\min}, \rho_{\max}]$ element-wisely. First, we easily see that h evaluated at $x_1 = x_2 = \cdots = x_n$ ranges in $(1, \infty)$.

To maximize the function h, we can simply maximize the function h over the subspace where $x_1 = \cdots = x_n$, since the maximum of h can be achieved in the subspace $x_1 = \cdots = x_n$. To maximize h, we can equivalently maximize the function $\frac{\sqrt{x}}{\sqrt{x+C}-\sqrt{C}}$ for $x \in [\rho_{\min}, \rho_{\max}]$, and it's easy to verify that the maximum is attained at $x = \rho_{\min}$. Moving back to upper bounding the greedy ratio at the ith iteration, we can get the following

$$\begin{aligned}\alpha_i &\leq \frac{\rho_{\min}^a}{[\rho_{\min} + (i-1)\rho_{\max}]^a - [(i-1)\rho_{\max}]^a} \\ &= \frac{1}{[1 + (i-1)\gamma]^a - [(i-1)\gamma]^a}\end{aligned}$$

□

Proof of Lemma 8

Proof. This Lemma immediately follows from Lemma 3. □