Submodular Cost Submodular Cover with an Approximate Oracle

Victoria G. Crawford [1] **Alan Kuhnle** [2] **My T. Thai** [1]

Abstract

In this work, we study the Submodular Cost Submodular Cover problem, which is to minimize the submodular cost required to ensure that the submodular benefit function exceeds a given threshold. Existing approximation ratios for the greedy algorithm assume a value oracle to the benefit function. However, access to a value oracle is not a realistic assumption for many applications of this problem, where the benefit function is difficult to compute. We present two incomparable approximation ratios for this problem with an approximate value oracle and demonstrate that the ratios take on empirically relevant values through a case study with the Influence Threshold problem in online social networks.

1. Introduction

Monotone[1] submodular set functions are found in many applications in machine learning and data mining (Kempe et al., 2003; Lin & Bilmes, 2011; Wei et al., 2013; Singla et al., 2016). A function $f : 2^S \to \mathbb{R}$ defined on subsets of a ground set S is submodular if for all $A \subseteq B \subseteq S$ and $x \notin B$,

$$f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B).$$

The ubiquity of submodular functions ensures that the optimization of submodular functions has received much attention (Nemhauser et al., 1978; Wolsey, 1982). In this work, we study the Submodular Cost Submodular Cover (SCSC) optimization problem, originally introduced by Wan et al. (2010) as a generalization of Wolsey (1982). SCSC is defined as follows.

[1]Department of Computer and Information Science and Engineering, University of Florida, Gainesville, Florida, United States [2]Department of Computer Science, Florida State University, Tallahassee, Florida, United States. Correspondence to: Victoria G. Crawford <vcrawford01@ufl.edu>, Alan Kuhnle <kuhnle@ufl.edu>, My T. Thai <mythai@cise.ufl.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

[1]For all $A \subseteq B \subseteq S$, $f(A) \leq f(B)$.

Submodular Cost Submodular Cover (SCSC) Let $f, c : 2^S \to \mathbb{R}_{\geq 0}$[2] be monotone submodular functions defined on subsets of a ground set S of size n. Given threshold $\tau \leq f(S)$, find argmin$\{c(X) | X \subseteq S, f(X) \geq \tau\}$.

Applications of SCSC include influence in social networks (Goyal et al., 2013; Kuhnle et al., 2017), data summarization (Mirzasoleiman et al., 2015; 2016), active set selection (Norouzi-Fard et al., 2016), recommendation systems (Guillory & Bilmes, 2011), and monitor placement (Soma & Yoshida, 2015; Zhang et al., 2016).

Existing approximations (Wolsey, 1982; Wan et al., 2010; Soma & Yoshida, 2015) to the NP-hard SCSC problem assume value oracle access to f, meaning that f can be queried at any subset $X \subseteq S$. Unfortunately, for many emerging applications of submodular functions, the function f is difficult to compute. Instead of having access directly to f, we may query only a surrogate function F that is ϵ-approximate to f, meaning that for all $X \subseteq S$

$$|f(X) - F(X)| \leq \epsilon.$$

For example, f may be approximated by a sketch (Badanidiyuru et al., 2012; Cohen et al., 2014), evaluated under noise (Chen et al., 2015; Singla et al., 2016), estimated via simulation (Kempe et al., 2003), or approximated by a learned function (Balcan et al., 2012).

If the surrogate function F is monotone submodular, then we may use existing approximation results for SCSC (see Appendix A). However, it is not always the case that the surrogate F maintains these properties. For example, the approximate influence oracle of Cohen et al. (2014) described in detail in Section 3 is non-submodular. To the best of our knowledge, no approximation results currently exist for the SCSC problem under a general approximate oracle.

1.1. Our Contributions

We provide an approximation ratio for SCSC if the greedy algorithm (Algorithm 1) has a value oracle to an ϵ-approximate function to f, provided that the smallest marginal gain μ[3] of any element that was added to the greedy

[2]We also assume that $f(\emptyset) = 0$, and $c(X) = 0$ if and only if $X = \emptyset$.

[3]See notation in Section 1.3 for a definition.

solution is sufficiently large relative to ϵ (Theorem 1). Our proof of Theorem 1 is a novel adaptation of the charging argument developed by Wan et al. (2010) for SCSC with integral-valued f, and has potential to be used for other versions of SCSC where f cannot be evaluated for all $X \subseteq S$. If the oracle error $\epsilon = 0$, our ratio nearly reduces to existing ratios for SCSC (Wolsey, 1982; Wan et al., 2010; Soma & Yoshida, 2015).

We provide a second, incomparable approximation ratio for SCSC if the greedy algorithm has a value oracle to an ϵ-approximate function to f, under the same conditions on the marginal gain as Theorem 1 (Theorem 2). In practical scenarios, the ratio of Theorem 1 is sometimes difficult to compute or bound because it requires evaluation of f. In contrast, an upper bound on the ratio of Theorem 2 is easy to compute only having access to the surrogate F. If the oracle error $\epsilon = 0$, our ratio is a new approximation ratio for SCSC that is incomparable to existing ratios for SCSC (Wolsey, 1982; Wan et al., 2010; Soma & Yoshida, 2015).

We demonstrate that the ratios of Theorem 1 and Theorem 2 take on empirically relevant values through a case study with the Influence Threshold (IT) problem under the independent cascade model on real social network datasets. This problem is a natural example of SCSC in which the function f is the expected activation of a seed set and is $\#P$-hard to compute, so optimization must proceed with a suitable surrogate function F. We use the average reachability sketch proposed by Cohen et al. (2014) for F, which is a non-submodular ϵ-approximation of f.

Organization In Section 1.2, we present an overview of related work on SCSC and submodular optimization with approximate oracles. Definitions used throughout the paper are presented in Section 1.3. The main approximation results (Theorems 1 and 2) are presented in Section 2. We consider the special case where F is monotone submodular in Appendix A. Finally, in Section 3, we compute the values that the approximation ratios of Theorems 1 and 2 take on for Influence Threshold.

1.2. Related Work

Submodular Cost Submodular Cover (SCSC) Approximation guarantees of the greedy algorithm with value oracle access to f for SCSC have previously been analyzed (Wolsey, 1982; Wan et al., 2010; Soma & Yoshida, 2015).

If c is modular[4] and f is integral valued, then Wolsey (1982) proved that the approximation ratio of the greedy algorithm is $\ln(\alpha)$, where α is the largest singleton value of f[5]. This is the best we can expect: set cover, which is a special case of SCSC, cannot be approximated within $(1 - \epsilon)\ln(n)$ unless NP has $n^{\mathcal{O}(\log(\log(n)))}$-time deterministic algorithms (Feige, 1998).

If f is real-valued, Wolsey proved that the greedy algorithm has an approximation ratio of $1 + \ln(\alpha/\beta)$, where β is the smallest non-zero marginal gain of adding any element to the greedy solution at any iteration[6]. In comparison, when the cost is modular and the value oracle exact ($\epsilon = 0$) then the ratio that we provide in Theorem 1 reduces to $2 + \ln(\alpha/\beta)$.

If f is integral, Wan et al. (2010) proved that the greedy algorithm has an approximation ratio of $\rho \ln(\alpha)$, where ρ is the curvature of c. Wan et al. developed a charging argument in order to deal with the general monotone submodular cost function. In the argument of Wan et al., the cost of the greedy solution, $c(A)$, is split up into charges over the elements of the optimal solution A^*. This method of charging will not work for SCSC with an ϵ-approximate oracle. This is because the elements chosen by the surrogate F do not necessarily exhibit diminishing cost-effectiveness. But, our argument is inspired by that of Wan et al.. Portions of our argument that share significant overlap with that of Wan et al. are made clear and restricted to the appendix. When f is integral and the value oracle exact ($\epsilon = 0$), the ratio that we provide in Theorem 1 reduces to $\rho(2 + \ln(\alpha))$.

Soma & Yoshida (2015) generalized SCSC to functions on the integer lattice, an extension of set functions. Soma & Yoshida proved that a decreasing threshold algorithm has a bicriteria approximation ratio of $(1 + 3\delta)\rho(1 + \ln(\alpha/\beta))$ to SCSC on the integer lattice, where $\delta < 1$ is an input. When the value oracle is exact ($\epsilon = 0$), the ratio that we provide in Theorem 1 reduces to $\rho(2 + \ln(\alpha/\beta))$.

The special case of SCSC where c is cardinality is the Submodular Cover (SC) problem. Distributed algorithms (Mirzasoleiman et al., 2015; 2016) as well as streaming algorithms (Norouzi-Fard et al., 2016) for SC have been developed and their approximation guarantees analyzed.

To the best of our knowledge, we are the first to study SCSC with an approximate oracle to f.

Optimization with Approximate Oracles A related problem to SCSC is Submodular Maximization (SM) with a cardinality constraint[7]. SM with a cardinality constraint and an approximate oracle has previously been analyzed (Horel & Singer, 2016; Qian et al., 2017).

Horel & Singer (2016) considered SM with a cardinality constraint where we seek to maximize F which is a rela-

[4] c is modular if for all $X \subseteq S$, $c(X) = \sum_{x \in X} c(\{x\})$.

[5] $\alpha = max_{x \in S} f(\{x\})$

[6] See notation in Section 1.3 for a definition.

[7] Given a budget κ and a monotone submodular function f defined on subsets of a ground set S of size n, find argmax$\{f(X) : |A| \leq \kappa\}$.

tive[8] ϵ-approximation to a monotone submodular function f. Under certain conditions on the oracle error, Horel & Singer found that the greedy algorithm yields a tight approximation ratio of $1 - 1/e - \mathcal{O}(\delta)$. On the other hand, Horel & Singer proved for any fixed $\beta > 0$, no algorithm given access to a $1/n^{1/2-\beta}$-approximate F can have a constant approximation ratio using polynomially many queries to F. The results of Horel & Singer can easily be translated into maximization of f with an approximate oracle. Another recent work for SM with a cardinality constraint and an approximate oracle is Qian et al. (2017), where a Pareto algorithm is analyzed.

A dual problem to SCSC is Submodular Cost Submodular Knapsack (SCSK)[9]. In the absence of oracle error, the approximation guarantees of SCSC and SCSK are connected (Iyer & Bilmes, 2013). In particular, Iyer & Bilmes proved that an (α, δ)-bicriteria[10] approximation algorithm for SCSK can be used to get a $((1+\epsilon)\delta, \alpha)$-bicriteria approximation algorithm for SCSC, and a similar result holds for the opposite direction.

Thus, the approximation results for SM with an approximate oracle could potentially be translated into an approximation guarantee for SCSC with an approximate oracle. However, the results of Horel & Singer are for the special case of SCSK where the cost function is cardinality. To the best of our knowledge, we are the first to study submodular optimization with oracle error and general monotone submodular cost functions. In addition, the feasibility guarantee provided under the Iyer & Bilmes framework is roughly $f(A) > (1 - 1/e)\tau$, so only a fraction of the threshold τ: in our work, we obtain the feasibility $f(A) \geq \tau - \epsilon$.

SM with an approximate oracle has been studied under different models for the surrogate F than an ϵ-approximation (Hassidim & Singer, 2017; Singer & Hassidim, 2018). These models are suited to applications where the approximate oracle is due to noise.

An approximate oracle to a submodular function results in a non-submodular function. Other works optimizing non-submodular or weakly submodular functions include Bian et al. (2017); Chen et al. (2018); Kuhnle et al. (2018).

1.3. Definitions

The definitions presented in this section are used throughout the paper.

[8]See discussion in Section 1.3 about ϵ-approximation.

[9] Given a budget κ and monotone submodular functions f, c defined on subsets of a ground set S of size n, find argmax$\{f(X) : c(A) \leq \kappa\}$.

[10]See Section 1.3 for a discussion of bicriteria approximation guarantees.

Algorithm 1 `greedy`(F, c, τ)

Input: A value oracle to $F : 2^S \to \mathbb{R}_{\geq 0}$, a value oracle to $c : 2^S \to \mathbb{R}_{\geq 0}$, and τ.
$F_\tau = \min\{F, \tau\}$
$A = \emptyset$
while $F(A) < \tau$ **do**
  $u = \text{argmax}_{x \in S \setminus A} \Delta F_\tau(A, x)/c(x)$
  $A = A \cup \{u\}$
end while
return A

Notation Given a function $g : 2^S \to \mathbb{R}_{\geq 0}$, define $g_\tau : 2^S \to \mathbb{R}_{\geq 0}$ to be $g_\tau(X) = \min\{\tau, g(X)\}$ for all $X \subseteq S$. In addition, we shorten the notation for marginal gain to be $\Delta g(X, x) = g(X \cup \{x\}) - g(X)$.

Given an instance of SCSC with cost function c and benefit function f, we define $c_{min} = \min_{x \in S} c(x)$, $c_{max} = \max_{x \in S} c(x)$, $\alpha = \max_{x \in S} f(\{x\})$, and ρ to be the curvature of c.

Suppose at the end of a run of Algorithm 1 there were k iterations of the while loop. Then we let A_i be A at the end of iteration $i \in \{1, ..., k\}$, $A_0 = \emptyset$, $\mu = \min\{f_\tau(A_i) - f_\tau(A_{i-1}) : i \in \{1, ..., k\}\}$, and $\beta = \min\{\Delta f_\tau(A_i, x) : i \in \{0, ..., k\}, x \in S, \Delta f_\tau(A_i, x) > 0\}$.

Greedy algorithm Pseudocode for the greedy algorithm that we analyze in Section 2 for SCSC with an ϵ-approximate oracle is given in Algorithm 1. Notice that when choosing an element at each iteration, we compute the marginal gain of F_τ and not F. Algorithm 1 was analyzed by Wan et al. for SCSC when a value oracle to f is given.

ϵ-Approximate A function $F : 2^S \to \mathbb{R}_{\geq 0}$ is ϵ-approximate to f if for all $X \subseteq S$, $|f(X) - F(X)| \leq \epsilon$. Notice that we use ϵ-approximate in an absolute sense, in contrast to ϵ-approximate in a relative sense: for all $X \subseteq S$, $|f(X) - F(X)| \leq \epsilon f(X)$. The latter is particularly useful if we are uncertain what range f takes on, in which case it is difficult to make meaningful requirements for additive noise.

Approximation in the relative sense can be converted into approximation in the absolute sense. Suppose F is an ϵ-approximation to f in the relative sense. If B is an upper bound on f, then F is an ϵB-approximation to f in the absolute sense. Over the duration of Algorithm 1, we can assume without loss of generality that τ is an upper bound on f.

Curvature The approximation guarantees presented in our work will use the curvature of the cost function c, as has been previously done for SCSC (Wan et al., 2010; Soma

& Yoshida, 2015). The curvature measures how modular[11] a function is. The curvature ρ of c is defined as $\rho = \max_{X \subseteq S} \sum_{x \in X} c(x)/c(X)$. If c is modular, $\rho = 1$, otherwise $\rho > 1$ (since c is submodular).

Bicriteria Approximation Algorithm We show in Section 2 that Algorithm 1 is a bicriteria approximation algorithm to SCSC, under certain conditions. A bicriteria approximation algorithm approximates the feasibility constraint ($f(A) \geq \tau$) in addition to the objective (minimize c). In our case, the feasibility guarantee is $f(A) \geq \tau - \epsilon$ if we have an ϵ-approximate oracle.

2. Approximation Results

In this section, we analyze the approximation guarantee of the greedy algorithm (Algorithm 1) for SCSC with an ϵ-approximate oracle. Definitions used in this section can be found in Section 1.3.

We first give a formal statement and discussion of our ratios in Section 2.1. In Section 2.2, we prove that the two ratios presented in Section 2.1 are incomparable. A sketch of the proofs of our results is presented in Section 2.3. Parts of the proofs not included in Section 2.3 are in Appendices B and C. The approximation guarantee of the greedy algorithm for SCSC with an ϵ-approximate oracle for the special case where the oracle is monotone submodular is in Appendix A.

2.1. Approximation Guarantees of the Greedy Algorithm for SCSC with an ϵ-Approximate Oracle

We present two approximation guarantees of the greedy algorithm (Algorithm 1) for SCSC with an ϵ-approximate oracle in Theorem 1 and Theorem 2. The guarantee in Theorem 1 corresponds more closely to existing approximation guarantees of SCSC (Wolsey, 1982; Wan et al., 2010; Soma & Yoshida, 2015) than that of Theorem 2. However, in some cases, Theorem 2 is easier to bound above. In general, the two ratios are incomparable; that is, there exist instances of SCSC where each dominates the other, as shown in Section 2.2.

Theorem 1. *Suppose we have an instance of SCSC with optimal solution A^*. Let F be a function that is ϵ-approximate to f.*

Suppose we run Algorithm 1 with input F, c, and τ. Then $f(A) \geq \tau - \epsilon$. And if $\mu > 4\epsilon c_{max}\rho/c_{min}$,

$$c(A) \leq \frac{\rho}{1 - \frac{4\epsilon c_{max}\rho}{c_{min}\mu}} \left(\ln\left(\frac{\alpha}{\beta}\right) + 2 \right) c(A^*).$$

[11] c is modular if $c(X) = \sum_{x \in X} c(\{x\})$.

Discussion of Theorem 1 In order for the ratio of Theorem 1 to hold, ϵ must be small enough relative to μ so that $\mu > 4\epsilon c_{max}\rho/c_{min}$. The lower bound on μ is used to upper bound the error introduced by choosing elements with F instead of f in the proof of Theorem 1 (see Section 2.3). Alternatively, we may ensure the approximation ratio of Theorem 1 as long as ϵ is sufficiently small by exiting Algorithm 1 if $F_\tau(A_{i+1}) - F_\tau(A_i)$ falls below an input value. The feasibility guarantee is weakened since Algorithm 1 does not necessarily run to completion, but not by much if ϵ is sufficiently small. The details of this alternative approximation guarantee can be found in Appendix A.

If $\epsilon = 0$, i.e. we have an oracle to the benefit function f, then the lower bound on μ is always satisfied and the approximation ratio in Theorem 1 nearly reduces to the ratio of previous approximation ratios for SCSC (Wolsey, 1982; Wan et al., 2010; Soma & Yoshida, 2015). In particular, the approximation ratio reduces to $\rho(\ln(\alpha/\beta) + 2)$. Compare this to the approximation ratio of Soma & Yoshida: $\rho(1 + 3\delta)(\ln(\alpha/\beta) + 1)$ where δ is an input that is greater than 0.

Computing α, β and μ in Theorem 1 requires evaluation of f. It is therefore of interest whether an upper bound can be computed on the ratio in Theorem 1 for a given instance of SCSC and a solution provided by the greedy algorithm, considering that we only have an oracle to F. We assume that the curvature ρ of c can be computed and focus on the values related to f. Without an oracle to f, the value α in Theorem 1 cannot be computed exactly, but α can be bounded above by using the oracle to F. Similarly, μ and β must be bounded below by using the oracle to F. However, a positive lower bound on β is problematic since it can be especially small and fall below the oracle error. This motivates our second approximation ratio, Theorem 2.

Theorem 2. *Suppose we have an instance of SCSC with optimal solution A^*. Let F be a function that is ϵ-approximate to f.*

Suppose we run Algorithm 1 with input F, c, and τ. Then $f(A) \geq \tau - \epsilon$. And if $\mu > 4\epsilon c_{max}\rho/c_{min}$, then for any $\gamma \in (0, 1 - 4\epsilon c_{max}\rho/c_{min}\mu)$,

$$c(A) \leq \frac{\rho}{1 - \frac{4\epsilon c_{max}\rho}{c_{min}\mu} - \gamma} \left(\ln\left(\frac{n\alpha\rho}{\gamma\mu}\right) + 2 \right) c(A^*).$$

Discussion of Theorem 2 If $\epsilon = 0$, i.e. we have an oracle to the benefit function f, then the lower bound on μ is always satisfied and the approximation ratio in Theorem 2 is a new approximation ratio for SCSC that is incomparable to those existing (Wolsey, 1982; Soma & Yoshida, 2015).

In contrast to the approximation guarantee of Theorem 1, the instance-dependent β has been replaced by γ in the approximation guarantee of Theorem 2. Since we no longer need a positive lower bound on β, this ratio was easy to bound in Section 3 by using the oracle to F. Also, since β is related to the minimum marginal gain on an instance, in some sense this ratio is more robust to the presence of very small marginal gains.

2.2. Incomparability of Guarantees of Theorem 1 and Theorem 2

In this section, we give examples that show that the approximation guarantees of Theorem 1 and of Theorem 2 are incomparable; for each guarantee, there exists an instance of SCSC where that guarantee is better than the other.

Examples We consider an instance of the Influence Threshold (IT) problem as defined in Appendix D under the independent cascade model of influence (Kempe et al., 2003).

Let $n \geq 3$. We construct a graph where $n-2$ vertices are in a clique, and all edge weights within the clique have weight 1. One remaining vertex is connected to the clique by an edge of weight $\sigma \in (0,1)$, and the other has degree 0.

Suppose we have an instance of SCSC where c is cardinality and $\tau = n-1+\sigma$. If we run Algorithm 1 with input f, c, τ, Algorithm 1 will return a single vertex from the clique and then the vertex with degree 0. In addition, $\alpha = n-2+\sigma$, $\mu = 1$, and $\beta = 1-\sigma$. Therefore the ratio from Theorem 1 is

$$\ln\left(\frac{n-2+\sigma}{1-\sigma}\right)+2 \tag{1}$$

and the ratio from Theorem 2 is for any $\gamma \in (0,1)$

$$\frac{1}{1-\gamma}\ln\left(\frac{n(n-2+\sigma)}{\gamma}\right)+2 \tag{2}$$

If we choose σ sufficiently close to 1, ratio (1) gets arbitrarily large; hence, there exists some γ where ratio (2) is smaller than ratio (1). On the other hand, as σ approaches 0, ratio (1) approaches $\ln(n-2)+2$. However, for any $\gamma \in (0,1)$, ratio (2) is at least $\ln(n(n-2))+2$.

2.3. Proof Sketches of Theorem 1 and 2

In this section, we present a sketch of the proofs of Theorem 1 and Theorem 2. The full proof for Theorem 1 and for Theorem 2 can be found in Appendix B and Appendix C respectively. Recall that definitions can be found in Section 1.3 and notation in Section 2.1.

Proof Sketch of Theorem 1 The feasibility guarantee is clear from the stopping condition on the greedy algorithm: $f(A) \geq F(A) - \epsilon \geq \tau - \epsilon$.

We now prove the upper bound on $c(A)$ if $\mu > 4\epsilon c_{max}\rho/c_{min}$. Without loss of generality we re-define $f = \min\{f, \tau\}$ and $F = \min\{F, \tau\}$. This way, $f = f_\tau$ and $F = F_\tau$. Notice that this does not change that F is an ϵ-approximation of f.

Let $x_1, ..., x_k$ be the elements of A in the order that they were chosen by Algorithm 1. If $k = 0$, then $c(A) = 0$ and the approximation ratio is clear. For the rest of the proof, we assume that $k \geq 1$.

We define a sequence of elements $\tilde{x}_1, ..., \tilde{x}_k$ where

$$\tilde{x}_i = \text{argmax}_{x \in S \setminus A_{i-1}} \frac{\Delta f(A_{i-1}, x)}{c(x)}.$$

$\tilde{x}_i$ has the most cost-effective marginal gain of being added to A_{i-1} according to f, while x_i has the most cost-effective marginal gain of being added to A_{i-1} according to F. Note that the same element can appear multiple times in the sequence $\tilde{x}_1, ..., \tilde{x}_k$. In addition, we have the following lower bound on $\Delta f(A_{i-1}, \tilde{x}_i)$:

$$\Delta f(A_{i-1}, \tilde{x}_i) \geq \frac{c(\tilde{x}_i)}{c(x_i)}\Delta f(A_{i-1}, x_i) \geq \frac{c_{min}}{c_{max}}\mu. \tag{1}$$

Our argument to bound $c(A)$ will follow the following three steps: **(a)** We bound $c(A)$ in terms of the costs of the elements $\tilde{x}_1, ..., \tilde{x}_k$. **(b)** We charge the elements of A^* with the costs of the elements $\tilde{x}_1, ..., \tilde{x}_k$, and bound $c(A)$ in terms of the total charge on all elements in A^*. **(c)** We bound the total charge on the elements of A^* in terms of $c(A^*)$.

(a) First, we bound $c(A)$ in terms of the costs of the elements $\tilde{x}_1, ..., \tilde{x}_k$. At iteration i of Algorithm 1, the most cost-effective element to add to A_{i-1} according to F is x_i. Using the fact that F is ϵ-approximate to f, we can bound how much more cost-effective $\tilde{x}_i$ is compared to x_i according to f as follows:

$$\frac{c(\tilde{x}_i)}{\Delta f(A_{i-1}, \tilde{x}_i)} + \alpha_i \geq \frac{c(x_i)}{\Delta f(A_{i-1}, x_i)} \tag{2}$$
$$\text{where } \alpha_i = \frac{2\epsilon(c(x_i) + c(\tilde{x}_i))}{\Delta f(A_{i-1}, \tilde{x}_i)\Delta f(A_{i-1}, x_i)}.$$

Inequality (2) and the submodularity of c imply that

$$\begin{aligned} c(A) &\leq \sum_{i=1}^{k} c(x_i) = \sum_{i=1}^{k} \Delta f(A_{i-1}, x_i)\frac{c(x_i)}{\Delta f(A_{i-1}, x_i)} \\ &\leq \sum_{i=1}^{k} \Delta f(A_{i-1}, x_i)\left(\frac{c(\tilde{x}_i)}{\Delta f(A_{i-1}, \tilde{x}_i)} + \alpha_i\right). \end{aligned} \tag{3}$$

We now bound the second term on the right side of Equation

(3) by

$$\sum_{i=1}^{k}\Delta f(A_{i-1},x_i)\alpha_i = \sum_{i=1}^{k}\frac{2\epsilon(c(\tilde{x}_i)+c(x_i))}{\Delta f(A_{i-1},\tilde{x}_i)} \leq \sum_{i=1}^{k}\frac{2\epsilon c(x_i)}{\Delta f(A_{i-1},x_i)}+\frac{2\epsilon c(x_i)}{\Delta f(A_{i-1},\tilde{x}_i)} \leq \frac{4\epsilon c_{max}}{c_{min}\mu}\sum_{i=1}^{k}c(x_i) \leq \frac{4\epsilon c_{max}\rho}{c_{min}\mu}c(A).$$

Applying this bound to (3) gives us the following bound on $c(A)$ in terms of the costs of the elements $\tilde{x}_1, ..., \tilde{x}_k$:

$$\left(1-\frac{4\epsilon c_{max}\rho}{c_{min}\mu}\right)c(A) \leq \sum_{i=1}^{k}\frac{\Delta f(A_{i-1},x_i)}{\Delta f(A_{i-1},\tilde{x}_i)}c(\tilde{x}_i). \quad (4)$$

(b) Next, we charge the elements of A^* with the costs of the elements $\tilde{x}_1, ..., \tilde{x}_k$, and bound $c(A)$ in terms of the total charge on all elements in A^*. By this we mean that we give each $y \in A^*$ a portion of the total cost of the elements $\tilde{x}_1, ..., \tilde{x}_k$. In particular, we give each $y \in A^*$ a charge of $w(y)$, defined by

$$w(y) = \sum_{i=1}^{k}(\pi_i(y)-\pi_{i+1}(y))\omega_i, \text{ where } \omega_i = \frac{c(\tilde{x}_i)}{\Delta f(A_{i-1},\tilde{x}_i)},$$

$$\text{and } \pi_i(y) = \begin{cases}\Delta f(A_{i-1},y) & i \in \{1,...,k\} \\ \Delta f(A,y) & i = k+1\end{cases}.$$

Recall that $\Delta f(A_{i-1},\tilde{x}_i) > 0$ for all i by Equation (1), and so we can define ω_i as above. Wan et al. charged the elements of A^* with the cost of elements $x_1, ..., x_k$ analogously to the above. We charge with the cost of elements $\tilde{x}_1, ..., \tilde{x}_k$ because they exhibit diminishing cost-effectiveness, i.e. $\omega_i - \omega_{i-1} \geq 0$ for all $i \in \{1, ..., k\}$, which is needed to proceed with the argument. Because we choose $x_1, ..., x_k$ with F, which is not monotone submodular, $x_1, ..., x_k$ do not exhibit diminishing cost-effectiveness even if we replace f with F in the definition of $w(y)$ above.

Using Equation (4) and an argument similar to Wan et al., we may work out that

$$\left(1-\frac{4\epsilon c_{max}\rho}{\mu c_{min}}\right)c(A) \leq \sum_{y\in A^*}w(y)+\Delta f(A,y)\omega_k. \quad (5)$$

$f(A)$ is not necessarily τ since the stopping condition for Algorithm 1 is only that $F(A) \geq \tau$. In this case, if $\Delta f(A,y) \neq 0$ for $y \in A^*$ (which implies that $y \notin A_{k-1}$) then by the submodularity of f

$$\Delta f(A,y)\omega_k \leq \Delta f(A,y)\frac{c(y)}{\Delta f(A_{k-1},y)} \leq c(y).$$

Therefore we can bound $c(A)$ in terms of the total charge on all elements in A^*:

$$\left(1-\frac{4\epsilon c_{max}\rho}{\mu c_{min}}\right)c(A) \leq \sum_{y\in A^*}w(y)+\rho c(A^*). \quad (6)$$

(c) Now, we bound the total charge on the elements of A^* in terms of $c(A^*)$.

We first define a value ℓ_y for every $y \in A^*$. For each $y \in A^*$, if $\pi_1(y) = 0$ we set $\ell_y = 0$, otherwise ℓ_y is the value in $\{1, ..., k\}$ such that if $i \in \{1, ..., \ell_y\}$ then $\pi_i(y) > 0$, and if $i \in \{\ell_y + 1, ..., k\}$ then $\pi_i(y) = 0$. Such an ℓ_y can be set since f is monotone submodular. Then

$$\sum_{y\in A^*}w(y) = \sum_{y\in A^*}\sum_{i=1}^{\ell_y}(\pi_i(y)-\pi_{i+1}(y))\omega_i = \sum_{y\in A^*}c(y)\left(\ln(\pi_1(y))-\ln(\pi_{\ell_y}(y))+1\right) \leq \rho\left(\ln\left(\frac{\alpha}{\beta}\right)+1\right)c(A^*). \quad (7)$$

Finally, we combine inequality (6) and inequality (7) to see that

$$\left(1-\frac{4\epsilon c_{max}\rho}{c_{min}\mu}\right)c(A) \leq \rho\left(\ln\left(\frac{\alpha}{\beta}\right)+2\right)c(A^*).$$

If $\mu > (4\epsilon c_{max}\rho)/(c_{min})$, this completes the proof of the approximation guarantee in the theorem statement. □

Proof Sketch of Theorem 2 The argument for the proof of Theorem 2 is the same as Theorem 1, except for part **(c)**. In particular, we have gotten to the point of the proof of Theorem 1 where we have proven that

$$\left(1-\frac{4\epsilon c_{max}\rho}{c_{min}\mu}\right)c(A) \leq \sum_{y\in A^*}w(y)+\rho c(A^*). \quad (1)$$

Let $\lambda > 0$. Then we define a value m_y for every $y \in A^*$ that is similar to ℓ_y but for when $\pi_i(y)$ falls below λ: For each $y \in A^*$, if $\pi_1(y) \leq \lambda$ we set $m_y = 0$, otherwise m_y is the value in $\{1, ..., k\}$ such that if $i \in \{1, ..., m_y\}$ then $\pi_i(y) > \lambda$, and if $i \in \{m_y+1, ..., k\}$ then $\pi_i(y) \leq \lambda$. Such an m_y can be set since f is monotone submodular. We may then use a similar analysis as in the proof of Theorem 1 to see that

$$w(y) \leq c(y)\left(\ln\left(\frac{\alpha}{\lambda}\right)+1\right)+\sum_{i=m_y+1}^{k}(\pi_i(y)-\pi_{i+1}(y))\omega_i \leq c(y)\left(\ln\left(\frac{\alpha}{\lambda}\right)+1\right)+\frac{\lambda\rho}{\mu}c(A).$$

By combining Equations (1) and the bound on $w(y)$, we have that

$$\left(1 - \frac{4\epsilon c_{max}\rho}{c_{min}\mu} - \frac{\lambda\rho n}{\mu}\right) c(A) \leq \rho \left(\ln\left(\frac{\alpha}{\lambda}\right) + 2\right) c(A^*).$$

If we set $\lambda = (\gamma\mu)/(n\rho)$ we have the approximation ratio in the theorem statement. □

3. Application and Experiments

In this section, we compute the approximation ratios stated in Theorems 1 and 2 on instances of the Influence Threshold problem (IT), a special case of SCSC. We use the non-submodular approximate reachability oracle that has been proposed by Cohen et al. (2014).

Influence Threshold Problem (IT) Let $G = (V, E)$ be a social network where vertices V represents users and edges E represent social connections. Suppose that $G_1 = (V, E_1), ..., G_N = (V, E_N)$, where $E_i \subseteq E$, represent N instances of "alive" social connections. In an instance, activation of users in the social network starts from an initial seed set and then propagates across edges. $c : 2^V \to \mathbb{R}_{\geq 0}$ is a monotone submodular function that gives the cost of seeding a set of users. For $X \subseteq V$, $f(X)$ is the average number of reachable vertices from X over the N instances. Given threshold $\tau \leq f(V)$, the Influence Threshold (IT) problem is to find the seed set argmin$\{c(X)|X \subseteq V, f(X) \geq \tau\}$.

Our definition of IT follows the simulation-based model of influence, as opposed to directly using a model such as Independent Cascade (IC) and defining f as the expected number of influenced vertices (Kempe et al., 2003). The IC model is commonly approximated by the simulation-based model, since computing the expected influence under the IC model is $\#P$-hard (Chen et al., 2010). In addition, the simulation-based model approximates the IC model arbitrarily well by choosing sufficiently large N. For more details on approximation of the IC model by the simulation-based model, see Appendix D.

Variations of IT where c is cardinality (He et al., 2014; Dinh et al., 2014; Kuhnle et al., 2017) or modular (Goyal et al., 2013; Han et al., 2017) have been studied in the influence literature. Notice the difference between IT and the Influence Maximization (IM) problem[12] (Kempe et al., 2003; Li et al., 2017). To the best of our knowledge, our approximation results are the first for IT with a general monotone submodular cost function.

The Approximate Average Reachability Oracle of Cohen et al. In order to have value oracle access to f in the IT problem, the instances $G_1, ..., G_N$ must be stored. In addition, to compute $f(X)$ for $X \subseteq V$ the reachable vertices from X must be computed for each of the N instances. If N is large, this is not scalable to large influence instances (Cohen et al., 2014).

[12] Given a budget κ, IM is to determine the set A such that $c(A) \leq \kappa$ and $f(A)$ is maximized.

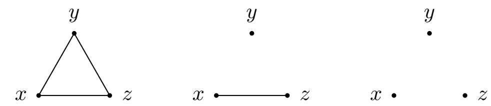


Figure 1. Instance G_1, G_2 and G_3 (in that order from left to right) from the proof of Proposition 1.

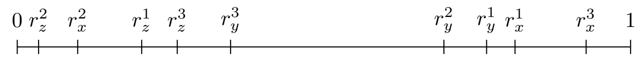


Figure 2. The mapping for each vertex, instance pair to the interval $[0, 1]$ for the proof of Proposition 1. r_u^i is the mapped value of vertex u, instance G_i.

Motivated by this, Cohen et al. (2014) proposed using an approximate average reachability oracle in place of f that is based on bottom-k min-hash sketches (Cohen, 1997). Given $k \in \mathbb{Z}_{>0}$, the approximate average reachability oracle F is constructed as follows: For every vertex, instance pair $(v, i) \in V \times \{1, ..., N\}$ a random rank value r_v^i is drawn from the uniform distribution on $[0, 1]$. For every vertex $u \in V$, the combined reachability sketch X_u of u is the smallest k values from the set $\{r_v^i : v \text{ is reachable from } u \text{ on instance } G_i\}$. X_u is stored for all $u \in V$. Note that generating X_u for all $u \in V$ does not require all the instances $G_1, ..., G_N$ to be stored at the same time.

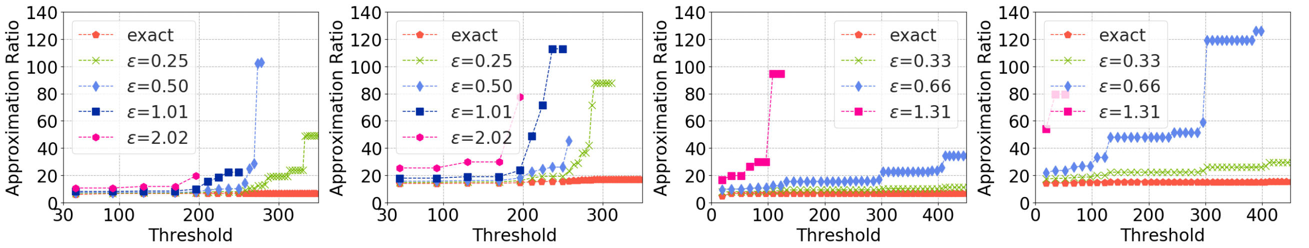
Let $X \subseteq S$. If $|\cup_{u \in X} X_u| < k$, then $F(X) = |\cup_{u \in X} X_u|/N$. Otherwise, let t be the k-th smallest value in $\cup_{u \in X} X_u$. Then $F(X) = (k-1)/(Nt)$.

F can be made an ϵ-approximation to f by choosing sufficiently large k: For $c > 2$, if $k = c\epsilon^{-2}\log(n)$ then the relative error of all queries over the duration of the greedy algorithm is within ϵ with probability at least $1 - 1/n^{c-2}$ (Cohen et al., 2014). The relative error can be converted to absolute error as described in Section 1.3.

Proposition 1. *The approximate average reachability oracle of Cohen et al. is non-submodular.*

Proof. Consider an instance of IT where $V = \{x, y, z\}$ and three instances G_1, G_2, G_3 are as depicted in Figure 1.

Suppose that we construct the approximate reachability oracle of Cohen et al. with $k = 5$, and the randomly generated mapping from vertex, instance pairs to the interval $[0, 1]$ is as depicted in Figure 2. Then $X_x = \{r_z^2, r_x^2, r_z^1, r_y^1, r_x^1\}$,



(a) facebook, r1 (b) facebook, r2 (c) GrQc, r1 (d) GrQc, r2

Figure 3. The approximation ratios of Theorem 1 (r1) and an upper bound on that of Theorem 2 (r2) at thresholds indicated by the markers.

$X_y = \{r_z^1, r_y^3, r_y^2, r_y^1, r_x^1\}$, and $X_z = \{r_z^2, r_x^2, r_z^1, r_z^3, r_y^1\}$. Consider adding z to the sets $A = \{x\}$ and $B = \{x, y\}$. Then

$$\Delta F(A, z) = (k-1)/(3r_y^1) - (k-1)/(3r_x^1),$$
$$\Delta F(B, z) = (k-1)/(3r_y^3) - (k-1)/(3r_y^2).$$

If $r_y^1 \approx r_x^1$, but r_y^3 is sufficiently smaller than r_y^2, $\Delta F(A, z) < \Delta F(B, z)$ despite $A \subseteq B$. □

The effectiveness of the approximate reachability oracle of Cohen et al. has been extensively evaluated for both IM (Cohen et al., 2014) as well as IT (Kuhnle et al., 2017), both with uniform cost. In particular, the approximate reachability oracle of Cohen et al. was demonstrated to be significantly faster than alternative approaches such as TIM (Tang et al., 2014).

Experimental Setup We use two real social networks: the Facebook ego network (Leskovec & Mcauley, 2012), and the ArXiV General Relativity collaboration network (Leskovec et al., 2007), which we refer to as GrQc. Influence propagation follows the Independent Cascade (IC) model (Kempe et al., 2003). The average reachability oracle, f, is over random realizations of the influence graph. The approximate average reachability oracle of Cohen et al., F, is computed over these realizations with various oracle errors ϵ and the greedy algorithm is run using these oracles. For comparison, we also run the greedy algorithm with f.

For the cost function, we choose a cost c_v for each $v \in V$ by sampling from a normal distribution and then define $c(X) = \sum_{x \in X} c_x$. We include additional experiments with non-modular cost functions in Appendix D. Additional details about the experimental setup can be found in Appendix D.

Approximation Ratio of the Greedy Algorithm As we described in Section 2.1, if the minimum marginal gain of any element added to the greedy solution is sufficiently large then Theorems 1 and 2 are approximation ratios.

For each run of the greedy algorithm with the approximate oracle F where the minimum marginal gain was sufficiently large, we compute the ratio of Theorem 1 exactly by querying f ("r1", Figures 3(a) and 3(c)) and we compute an upper bound on the ratio of Theorem 2 with only F using the argument described in the discussion of Theorem 2 ("r2", Figures 3(b) and 3(d)). Details of how we selected γ for the ratio of Theorem 2 can be found in Appendix D. We were generally unable to compute an upper bound on the ratio presented in Theorem 1 with just an oracle to F because the β term was too small.

In addition, we ran the greedy algorithm for the exact oracle f and computed the ratios of Theorem 1 and Theorem 2 exactly where $\epsilon = 0$. Note that the ratio of Theorem 1 when $\epsilon = 0$ reduces to the approximation ratio for SCSC given by Soma & Yoshida (2015).

Results The approximation ratios on the Facebook dataset are plotted in Figures 3(a) and 3(b). A marker indicates the threshold τ given as input to the greedy algorithm, and the corresponding ratio for that run. We chose the F values at each step of the greedy algorithm to be the thresholds. If the minimum marginal gain of a run was too small relative to ϵ for a threshold, or the approximation was greater than 140, then no marker is plotted.

As expected, smaller oracle error ϵ implies better approximation ratios. For the same ϵ, the ratio of Theorem 1 is better than the upper bound on the ratio of Theorem 2. But, recall that the latter can be computed only querying F. As the threshold increases, the ratio degrades because elements with smaller marginal gain are being added into the solution set, although the effect is smaller with smaller ϵ.

Qualitatively similar results are shown for the GrQc dataset in Fig. 3(c) and 3(d). The ratios degrade less quickly in GrQc compared to Facebook: this is because in Facebook there are a few vertices that can be selected with high marginal gain, and then after that the marginal gains quickly decrease. In contrast, the marginal gains for GrQc are more sustained over the course of the greedy algorithm.

4. Acknowledgements

This work was supported in part by NSF CNS-1814614, NSF EFRI 1441231, and DTRA HDTRA1-14-1-0055. Victoria G. Crawford was supported by a Harris Corporation Fellowship. We thank the anonymous reviewers for their helpful feedback.

References

Badanidiyuru, A., Dobzinski, S., Fu, H., Kleinberg, R., Nisan, N., and Roughgarden, T. Sketching valuation functions. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pp. 1025–1035. Society for Industrial and Applied Mathematics, 2012.

Balcan, M. F., Constantin, F., Iwata, S., and Wang, L. Learning valuation functions. In *Conference on Learning Theory*, pp. 4–1, 2012.

Bian, A. A., Buhmann, J. M., Krause, A., and Tschiatschek, S. Guarantees for Greedy Maximization of Non-submodular Functions with Applications. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

Chen, L., Feldman, M., and Karbasi, A. Weakly Submodular Maximization Beyond Cardinality Constraints: Does Randomization Help Greedy? In *International Conference on Machine Learning (ICML)*, 2018. URL http://arxiv.org/abs/1707.04347.

Chen, W., Wang, C., and Wang, Y. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1029–1038. ACM, 2010.

Chen, Y., Hassani, S. H., Karbasi, A., and Krause, A. Sequential information maximization: When is greedy near-optimal? In *Conference on Learning Theory*, pp. 338–363, 2015.

Cohen, E. Size-estimation framework with applications to transitive closure and reachability. *Journal of Computer and System Sciences*, 55(3):441–453, 1997.

Cohen, E., Delling, D., Pajor, T., and Werneck, R. F. Sketch-based influence maximization and computation: Scaling up with guarantees. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pp. 629–638. ACM, 2014.

Dinh, T. N., Zhang, H., Nguyen, D. T., and Thai, M. T. Cost-effective viral marketing for time-critical campaigns in large-scale social networks. *IEEE/ACM Transactions on Networking (ToN)*, 22(6):2001–2011, 2014.

Feige, U. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

Goyal, A., Bonchi, F., Lakshmanan, L. V., and Venkatasubramanian, S. On minimizing budget and time in influence propagation over social networks. *Social network analysis and mining*, 3(2):179–192, 2013.

Guillory, A. and Bilmes, J. A. Simultaneous learning and covering with adversarial noise. In *ICML*, volume 11, pp. 369–376, 2011.

Han, K., He, Y., Tang, S., Huang, H., and Xu, C. Cost-effective seed selection in online social networks. *arXiv preprint arXiv:1711.10665*, 2017.

Hassidim, A. and Singer, Y. Submodular optimization under noise. In *Conference on Learning Theory*, pp. 1069–1122, 2017.

He, J. S., Ji, S., Beyah, R., and Cai, Z. Minimum-sized influential node set selection for social networks under the independent cascade model. In *Proceedings of the 15th ACM International Symposium on Mobile ad hoc Networking and Computing*, pp. 93–102. ACM, 2014.

Horel, T. and Singer, Y. Maximization of approximately submodular functions. In *Advances in Neural Information Processing Systems*, pp. 3045–3053, 2016.

Iyer, R. K. and Bilmes, J. A. Submodular optimization with submodular cover and submodular knapsack constraints. In *Advances in Neural Information Processing Systems*, pp. 2436–2444, 2013.

Kempe, D., Kleinberg, J., and Tardos, É. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 137–146. ACM, 2003.

Kuhnle, A., Pan, T., Alim, M. A., and Thai, M. T. Scalable bicriteria algorithms for the threshold activation problem in online social networks. In *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*, pp. 1–9. IEEE, 2017.

Kuhnle, A., Smith, J. D., Crawford, V. G., and Thai, M. T. Fast Maximization of Non-Submodular, Monotonic Functions on the Integer Lattice. In *International Conference on Machine Learning (ICML)*, 2018. URL http://arxiv.org/abs/1805.06990.

Leskovec, J. and Mcauley, J. J. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*, pp. 539–547, 2012.

Leskovec, J., Kleinberg, J., and Faloutsos, C. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1 (1):2, 2007.

Li, X., Smith, J. D., Dinh, T. N., and Thai, M. T. Why approximate when you can get the exact? optimal targeted viral marketing at scale. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pp. 1–9. IEEE, 2017.

Lin, H. and Bilmes, J. 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*, pp. 510–520. Association for Computational Linguistics, 2011.

Mirzasoleiman, B., Karbasi, A., Badanidiyuru, A., and Krause, A. Distributed submodular cover: Succinctly summarizing massive data. In *Advances in Neural Information Processing Systems*, pp. 2881–2889, 2015.

Mirzasoleiman, B., Zadimoghaddam, M., and Karbasi, A. Fast distributed submodular cover: Public-private data summarization. In *Advances in Neural Information Processing Systems*, pp. 3594–3602, 2016.

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

Norouzi-Fard, A., Bazzi, A., Bogunovic, I., El Halabi, M., Hsieh, Y.-P., and Cevher, V. An efficient streaming algorithm for the submodular cover problem. In *Advances in Neural Information Processing Systems*, pp. 4493–4501, 2016.

Qian, C., Shi, J.-C., Yu, Y., Tang, K., and Zhou, Z.-H. Subset selection under noise. In *Advances in Neural Information Processing Systems*, pp. 3560–3570, 2017.

Singer, Y. and Hassidim, A. Optimization for approximate submodularity. In *Advances in Neural Information Processing Systems*, pp. 394–405, 2018.

Singla, A., Tschiatschek, S., and Krause, A. Noisy submodular maximization via adaptive sampling with applications to crowdsourced image collection summarization. In *AAAI*, pp. 2037–2043, 2016.

Soma, T. and Yoshida, Y. A generalization of submodular cover via the diminishing return property on the integer lattice. In *Advances in Neural Information Processing Systems*, pp. 847–855, 2015.

Tang, Y., Xiao, X., and Shi, Y. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 75–86. ACM, 2014.

Wan, P.-J., Du, D.-Z., Pardalos, P., and Wu, W. Greedy approximations for minimum submodular cover with submodular cost. *Computational Optimization and Applications*, 45(2):463–474, 2010.

Wei, K., Liu, Y., Kirchhoff, K., and Bilmes, J. Using document summarization techniques for speech data subset selection. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 721–726, 2013.

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

Zhang, H., Kuhnle, A., Zhang, H., and Thai, M. T. Detecting misinformation in online social networks before it is too late. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 541–548. IEEE Press, 2016.