Regularized Submodular Maximization at Scale

Ehsan Kazemi[1] Shervin Minaee[2] Moran Feldman[3] Amin Karbasi[4]

Abstract

In this paper, we propose scalable methods for maximizing a regularized submodular function f, expressed as the difference between a monotone submodular function g and a modular function ℓ. Submodularity is related to the notions of diversity, coverage, and representativeness. In particular, finding the mode (most likely configuration) of many popular probabilistic models of diversity, such as determinantal point processes and strongly log-concave distributions, involves maximization of (regularized) submodular functions. Since a regularized function can potentially take on negative values, the classic theory of submodular maximization, which heavily relies on a non-negativity assumption, is not applicable. We avoid this issue by developing the first one-pass streaming algorithm for maximizing a regularized submodular function subject to a cardinality constraint. Furthermore, we give the first distributed algorithm that (roughly) reproduces the guarantees of state-of-the-art centralized algorithms for the problem using only $O(1/\varepsilon)$ rounds of MapReduce. We highlight that our result, even for the unregularized case where the modular term ℓ is zero, improves over the memory and communication complexity of the state-of-the-art by a factor of $O(1/\varepsilon)$. We also empirically study the performance of our scalable methods on real-life applications, including finding the mode of negatively correlated distributions, vertex cover of social networks, and several data summarization tasks.

1. Introduction

Finding a diverse set of items, also known as data summarization, is one of the central tasks in machine learning. It usually involves either maximizing a utility function that promotes coverage and representativeness (Wei et al., 2015) (we call this an optimization perspective) or sampling from discrete probabilistic models that promote negative correlations and show repulsive behaviors (Gotovos et al., 2015) (we call this a sampling perspective). Celebrated examples of probabilistic models that encourage negative dependency include determinantal point processes (Kulesza & Taskar, 2012), strongly Rayleigh measures (Borcea et al., 2009) strongly log-concave distributions (Gurvits, 2009), and probabilistic submodular models (Djolonga & Krause, 2014; Iyer & Bilmes, 2015). In fact, the two above views are related in the sense that often the mode (most likely configuration) of a diversity promoting distribution is a simple variant of a (regularized) submodular function. For instance, determinantal point processes are log-submodular. Or, as we show later, a strongly log-concave distribution is a regularized log-submodular plus a log quadratic term. This paper aims to show how such optimization tasks can be done at scale.

From the optimization perspective, to effectively select a diverse subset of items, we need to define a measure that captures the representativeness that lies within a selected subset. Often, such a measure naturally has the diminishing returns property, and thus, is formally captured by *submodularity*. Given a finite ground set $\mathcal{N}$ of size n, consider a set function $g\colon 2^{\mathcal{N}} \to \mathbb{R}$ assigning a utility $g(A)$ to every subset $A \subseteq \mathcal{N}$. We say that g is submodular if for any pair of subsets $A \subseteq B \subseteq \mathcal{N}$ and element $u \notin B$, we have $g(A \cup \{u\}) - g(A) \geq g(B \cup \{u\}) - g(B)$, which intuitively means that the increase in "representativeness" following the addition of u to a set is smaller when u is added to a larger set. For shorthand, we write the marginal gain of an element as $g(e \mid S) \triangleq g(S \cup \{e\}) - g(S)$ and the marginal gain of adding an entire set as $g(A \mid S) \triangleq g(S \cup A) - g(S)$. Additionally, a set function is said to be monotone if $g(A) \leq g(B)$ for all $A \subseteq B \subseteq \mathcal{N}$; that is, adding more data can only increase the representativeness. Many of the previous works in data summarization and diverse subset selection that take an optimization perspective simply aim to maximize a monotone submodular function (Mirzasoleiman et al., 2013). Monotonicity has the advantage of promoting coverage, but it also enhances the danger of over-fitting to the data as adding more elements can never decrease the utility. To address this issue, as is often done in machine learning, we need to add a simple penalty, or a regular-

[1]Google, Zürich, Switzerland [2]Snap Inc [3]Department of Computer Science, University of Haifa, Israel [4]Yale Institute for Network Science, Yale University. Correspondence to: Ehsan Kazemi <ehsankazemi@google.com>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

izer, term. Formally, we get an instance of the **regularized submodular maximization** problem, which is defined as follows. Given an objective function $f(S) \triangleq g(S) - \ell(S)$, where g is a non-negative monotone submodular function and ℓ is a non-negative modular function, solve

$$\arg\max_{S \subseteq \mathcal{N}, |S| \leq k} f(S) = \arg\max_{S \subseteq \mathcal{N}, |S| \leq k} [g(S) - \ell(S)] \ , \qquad (1)$$

i.e., find a set S of size at most k maximizing the function f (A function $\ell \colon 2^{\mathcal{N}} \to \mathbb{R}$ is modular if there is a value ℓ_u for every $u \in \ell$ such that $\ell(S) = \sum_{u \in S} \ell_u$ for every $S \subseteq \mathcal{N}$.)

Example Applications. Strongly Rayleigh (SR) measures (Borcea et al., 2009) (including determinantal point processes (Kulesza & Taskar, 2012)) or the more general class of strongly log-concave (SLC) distributions (Gurvits, 2009) provide strong negative dependence among sampling items. Robinson et al. (2019) showed that the logarithm of an SLC distribution enjoys a variant of approximate submodularity referred to as γ-additively weak.[1] In Lemma 8, we show that a γ-additively weak submodular function can be roughly rewritten as the difference between a non-negative monotone submodular function and a modular function. Therefore, an efficient and scalable algorithm for maximizing regularized submodular functions could be instrumental for finding the mode of SLC distributions. Diversity is another trait of a good summary, and there are several ways to quantify it. In this regard, while submodularity is still quite a natural property, monotonicity sometimes is not (Tschiatschek et al., 2014). As a result, to encourage diversity, a natural way is to add a modular regularization term (Tschiatschek et al., 2014). Moreover, regularized submodular functions arise in cases with a required balance between coverage and cost such as team formation (Iyer & Bilmes, 2013; Ene et al., 2020).

Why Do We Need a New Theory? It is easy to show that $f \triangleq g - \ell$ is still a submodular function. However, **it may no longer be non-negative**, an assumption that is essential for deriving competitive algorithms with constant-factor approximation guarantees (for more information, see the survey by Buchbinder & Feldman (2018)). A natural way to bypass this issue is to shift the submodular function to make it non-negative, and then use an algorithm designed for optimizing a classic submodular maximization problem. More specifically, given an α-approximation algorithm for non-negative submodular function f and a shift C making it non-negative, the guarantee we get is $f(S) \geq \alpha \cdot f(OPT) - (1 - \alpha) \cdot C$, where OPT is an optimal solution for the problem. As can be seen from this forumla, the approximation guarantee obtained in this way depends on the size of shift, and unfortunately becomes useless when the necessary shift is large (which can be the case even when $\ell(OPT)$ is relatively small). We also note that, even though maximizing a regularized submodular function has been proposed in the past as a more faithful model of diverse data selection (Tschiatschek et al., 2014), a formal treatment of this problem has only recently been done (Sviridenko et al., 2017; Feldman, 2021; Harshaw et al., 2019).

[1]Note that strong log-concavity does not imply log-submodularity (Gotovos, 2019).

Streaming and Distributed Settings. In many practical scenarios, random access to the entire data is not possible as only a small fraction of the data can be loaded to the main memory and the data arrives at a very fast pace allowing it to be read only once. Furthermore, the amount of collected data is often too large to solve the optimization problem on a single machine. Indeed, with the unprecedented increase in the data size in recent years, scalable data summarization algorithms able to handle such scenarios have gained a lot of attention with far-reaching applications (Lin & Bilmes, 2011; Das & Kempe, 2011; Tschiatschek et al., 2014; Gygli et al., 2015; Elenberg et al., 2017; 2018; Feldman et al., 2018; Mitrovic et al., 2018; Kazemi et al., 2018; Haba et al., 2020).

Handling similar scenarios in our setting requires the use of streaming and distributed algorithms for Problem (1), but no such algorithms were known prior to this work. Based on ideas from (Sviridenko et al., 2017; Feldman, 2021), Harshaw et al. (2019) proposed DISTORTED-GREEDY, an efficient offline algorithm to (approximately) solve this problem. This algorithm iteratively and greedily finds elements that maximize a distorted objective function. As a centralized algorithm, DISTORTED-GREEDY requires a memory that grows linearly with the size of the data, and it needs to make multiple passes ($\Theta(n)$ in the worst case) over the data; therefore it fails to satisfy the above-mentioned requirements of modern applications.

Our Results. In the following, we briefly explain our main theoretical results. In Section 3, we introduce the first one-pass streaming algorithm (called THRESHOLD-STREAMING) for maximizing a regularized submodular function subject to a k-cardinality constraint, where the theoretical guarantee (see Theorem 4) depends on an input parameter $r > 0$ to the algorithm.[2] The value for r yielding the strongest guarantee for Problem (1) depends on the unknown ratio $g(OPT)/\ell(OPT)$, but we manage to simulate knowing it using an efficient guessing scheme (see Algorithm 2). Theorem 1 gives the guarantee that we obtain in this way. We note that most previous works did not work out the details of finding the best choice for their parameter corresponding to r, and just give the result for $r = 1$.

[2]Technically, this algorithm is a semi-streaming algorithm, as its space complexity is nearly linear in the size of the solution (rather than poly-logarithmic as in true streaming algorithms). Since this is unavoidable for algorithms that output the solution itself (rather than just estimate its value), we ignore the distinction between the two types of algorithms in this paper.

Theorem 1. *Given parameters $\varepsilon, \delta > 0$,* DISTORTED-STREAMING *(Algorithm 2) outputs a set S obeying*

$$g(S) - \ell(S) \geq ((1-\delta')\zeta_{OPT} - \varepsilon')\left(g(OPT) - \ell(OPT)\right),$$

where $\delta' = \delta/2$, $\varepsilon' = \frac{\varepsilon}{2\zeta_{OPT}}$ and ζ_{OPT} is

$$\frac{g(OPT) - \sqrt{\ell(OPT) \cdot [2g(OPT) - \ell(OPT)]}}{2[g(OPT) - \ell(OPT)]}.$$

In Fig. 1, we plot the approximation factor ζ_{OPT} as a function of $g(OPT)/\ell(OPT)$. We observe that, as $g(OPT)/\ell(OPT)$ grows, ζ_{OPT} approaches $1/2$. The last value is optimal for large values of $g(OPT)/\ell(OPT)$ since such values imply that the modular cost function ℓ is negligable compared to the utility function g, and Feldman et al. (2020) proved that the $1/2$ approximation ratio is optimal when ℓ is missing altogether. Based on this evidence, we **conjecture** that our streaming algorithm for (1) is optimal.

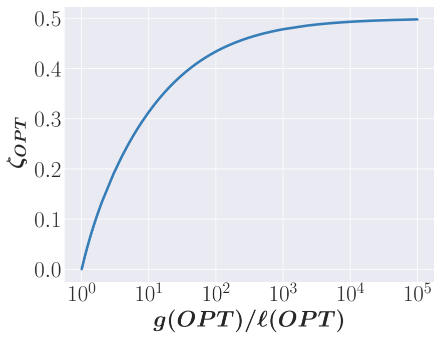


Figure 1. ζ_{OPT} as a function of $g(OPT)/\ell(OPT)$.

In Section 4, we develop DISTORTED-DISTRIBUTED, the first distributed algorithm for regularized submodular maximization in a MapReduce model of computation. This algorithm allows us to distribute data across several machines and use their combined computational resources. The approximation guarantee of DISTORTED-DISTRIBUTED is given in Theorem 2.[3]

Theorem 2. DISTORTED-DISTRIBUTED *(Algorithm 3) returns a set $D \subseteq \mathcal{N}$ of size at most k such that*

$$\mathbb{E}[g(D) - \ell(D)] \geq (1-\varepsilon)\left[(1-e^{-1})g(OPT) - \ell(OPT)\right].$$

Interestingly, even for the classic case of an unregulated monotone submodular function, **our technique improves over the space and communication complexity of the existing work (Barbosa et al., 2016) by a factor of $\Theta(1/\varepsilon)$ under a several constraints including: cardinality, matroids and p-systems.** Since stating most of the results of (Barbosa et al., 2016) requires many terms that are unrelated to the main topic of the current paper, we state here only one out of the multiple results that can be obtained by combining our technique with that of Barbosa et al. (2016).

[3]We should mention that it is possible to extend the idea we used in the streaming algorithm to the distributed scenario and provide the final guarantee for the distributed setting in the form of Theorem 1 (i.e., multiplicative approximation ratio of $g(OPT) - \ell(OPT)$ that depends on $g(OPT)/\ell(OPT)$). Alternatively, one can get a similar result by exploiting a recent result from (Feldman, 2021) which provides such a guarantee in the offline setting. We leave this extension to the future work.

Theorem 3. *Given a hereditary set system $(\mathcal{N}, \mathcal{I})$ of rank R (i.e., $R = \max_{S \in \mathcal{I}} |S|$). If the greedy algorithm obtains α-approximation for the problem of finding a set $S \in \mathcal{I}$ maximizing a given non-negative monotone submodular function $f : 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$, then, for every $\varepsilon > 0$ and number m of machines, there exists a MapReduce algorithm for this problem that (i) uses $\tilde{O}(\varepsilon^{-1})$ MapReduce rounds, (ii) has a space complexity of $O(|\mathcal{N}|/m + mR/\varepsilon)$ per machine (with high probability) and $\tilde{O}(|\mathcal{N}| + m^2R/\varepsilon)$ in total, and (iii) has an approximation ratio of $\alpha - \varepsilon$.*

In Section 5, we show how finding the mode of a class of strongly log-concave (SLC) distributions can be reduced to the maximization of a regularized submodular function. Finally, in Section 6 and Appendices D.2 and E, we explore the power of the regularized submodular maximization approach and our algorithms in several real-world applications through an extensive set of experiments. **Most of the proofs are deferred to the Supplementary Material.**

2. Related Work

Finding the optimal solution for a submodular maximization problem is computationally hard even in the absence of a constraint (Feige et al., 2011). However, a long list of works has suggested approximation algorithms for such problems with many kinds of constraints (see, e.g., (Nemhauser et al., 1978; Lee et al., 2010; Feldman et al., 2011; Badanidiyuru & Vondrák, 2014; Feldman et al., 2017)). Nevertheless, up until very recently, all the existing works required the objective function to take only non-negative values, an assumption that may not hold in many applications (Harshaw et al., 2019). The first work to handle submodular objective functions that might take negative values was done by Sviridenko et al. (2017), who studied the maximization of submodular functions that can be decomposed as a sum $g + c$, where g is a non-negative monotone submodular function and c is an (arbitrary) modular function. For this problem, Sviridenko et al. (2017) gave two randomized polynomial-time algorithms producing a set S roughly obeying $g(S) + c(S) \geq (1 - 1/e) \cdot g(OPT) + c(OPT)$. Both algorithms, however, are mainly of theoretical interest, as their computational complexity is prohibitive. Using ideas due to Feldman (2021), Harshaw et al. (2019) showed that in the case of a cardinality constraint (and a non-negative c) much of the complexity can be avoided, yielding the first practical algorithm for such functions, named DISTORTED-GREEDY. They also extended their results to γ-weakly submodular functions and the unconstrained setting.

Due to the massive volume of current data sets, scalable methods have gained a lot of interest in machine learning applications. One appealing approach towards this goal is to design streaming algorithms. Badanidiyuru et al. (2014) were the first to consider a single-pass streaming algorithm for maximizing a monotone submodular function under a cardinality constraint. Their result was later improved and extended to non-monotone functions (Alaluf et al., 2020; Kazemi et al., 2019) and more involved constraints (Buchbinder et al., 2015; Chekuri et al., 2015; Feldman et al., 2018). Another scalable approach is the development of distributed algorithms through the MapReduce framework, where the data is split amongst several machines and processed in parallel (Mirzasoleiman et al., 2016b; Barbosa et al., 2015; Mirrokni & Zadimoghaddam, 2015; Barbosa et al., 2016; Liu & Vondrák, 2019).

3. Streaming Algorithm

In this section, we present our proposed streaming algorithm for Problem (1). To explain our algorithm, let us first define T to be a subset of $\mathcal{N}$ of size at most k such that $T \in \arg\max_{S\subseteq\mathcal{N},|S|\leq k}[(h(r)-\varepsilon)\cdot g(S)-r\cdot\ell(S)]$, where r is some positive real value to be discussed later, and $h(r) = \frac{2r+1-\sqrt{4r^2+1}}{2}$. A basic version of our proposed algorithm, named THRESHOLD-STREAMING, is given as Algorithm 1. We note that this algorithm guesses, in the first step, a value $\tau > 0$ which obeys $k\tau \leq h(r)\cdot g(T)-r\cdot\ell(T) \leq (1+\varepsilon)k\tau$. In Algorithm 1, to avoid unnecessary technicalities, we simply assume that the algorithm can guess such a value. In Appendix B, we explain how a technique from (Badanidiyuru et al., 2014) can be used to implement this step at the cost of increasing the space complexity of the algorithm by a factor of $O(\varepsilon^{-1}(\log k + \log r^{-1}))$. Algorithm 1 starts with an empty set S. While the size of set S is still smaller than k, for every incoming element u, the value of $g(u \mid S) - \alpha(r)\cdot\ell(\{u\})$ is calculated, where $\alpha(r) = \frac{2r+1+\sqrt{4r^2+1}}{2}$. If this value is at least τ, then u is added to S by the algorithm. The theoretical guarantee of Algorithm 1 is provided in Theorem 4.[4]

Theorem 4. *For every $\varepsilon, r > 0$,* THRESHOLD-STREAMING *produces a set $S \subseteq \mathcal{N}$ of size at most k for Problem (1), obeying $g(S)-\ell(S) \geq \max_{T\subseteq\mathcal{N},|T|\leq k}[(h(r)-\varepsilon)\cdot g(T) - r\cdot\ell(T)]$, where $h(r) = \frac{2r+1-\sqrt{4r^2+1}}{2}$.*

[4]The formulas for $h(r)$ and $\alpha(r)$ are, unfortunately, quite unintuitive. However, it is interesting to note that the ratio $h(r)/r$ between the coefficients of g and ℓ in the guarantee of Theorem 4 is equal to the ratio $1/\alpha(r)$ between the coefficients of these two functions in Line 6 of Algorithm 1.

We should point out that previous studies of Problem (1), for various theoretical and practical reasons, have only focused on the case in which $r = 1$ and T is the set OPT of size at most k maximizing $g(T) - \ell(T)$ (Sviridenko et al., 2017; Harshaw et al., 2019). In this case, we get the following corollary from the result of Theorem 4.

Corollary 5. *For every $\varepsilon > 0$,* THRESHOLD-STREAMING *produces a set $S \subseteq \mathcal{N}$ of size at most k for Problem (1) obeying $g(S) - \ell(S) \geq (\phi^{-2} - \varepsilon)\cdot g(OPT) - \ell(OPT)$, where ϕ is the golden ratio (and thus, $\phi^{-2} \approx 0.382$).*

Algorithm 1: THRESHOLD-STREAMING

```
1 Guess a value $\tau$ such that
    $k\tau \leq h(r)\cdot g(T) - r\cdot\ell(T) \leq (1+\varepsilon)k\tau$.
2 Let $\alpha(r) \leftarrow \frac{2r+1+\sqrt{4r^2+1}}{2}$.
3 Let $S \leftarrow \varnothing$.
4 while $|S| < k$ and there are more elements do
5     Let $u$ be the next elements in the stream.
6     if $g(u \mid S) - \alpha(r)\cdot\ell(\{u\}) \geq \tau$ then
7         Add $u$ to the set $S$.
8 return the better solution among $S$ and $\varnothing$.
```

Next, we study the effect of the parameter r on the performance of Algorithm 1 under different settings. First note that the bound given by Corollary 5 reduces to a trivial lower bound of 0 when $\phi^{-2}\cdot g(OPT) \leq \ell(OPT)$. A similar phenomenon happens for the bound of (Harshaw et al., 2019, Theorem 3) when $(1-e^{-1})\cdot g(OPT) \leq \ell(OPT)$. Namely, in this regimen their bound (i.e., $(1-e^{-1})\cdot g(OPT) - \ell(OPT)$) becomes trivial. We now explain how a carefully chosen value for r can be used to prevent this issue.

For a set S, let β_S denote the ratio between the utility of S and its linear cost, i.e., $\beta_S = \frac{g(S)-\ell(S)}{\ell(S)}$. Using this terminology, we get that the guarantees of Corollary (5) and (Harshaw et al., 2019, Theorem 3) become trivial when $\beta_{OPT} \leq \phi^2 - 1 = \phi$ and $\beta_{OPT} \leq 1/(e-1)$, respectively. In Corollary 6, we show that by knowing the value of β_{OPT} we can find a value for r in Algorithm 1 which makes Theorem 4 yield the strongest guarantee for (1), and moreover, this guarantee is non-trivial as long as $\beta_{OPT} > 0$ (if $\beta_{OPT} \leq 0$, then the empty set is a trivial optimal solution).[5]

Corollary 6. *Assume the value of β_{OPT} is given, and let $\varepsilon' = \varepsilon\cdot(1 + 1/\beta_{OPT})$. Setting $r = r_{OPT} = \frac{\beta_{OPT}}{2\sqrt{1+2\beta_{OPT}}}$*

[5]The value β_{OPT} is undefined when $\ell(OPT) = 0$. We implicitly assume in this section that this does not happen. However, if this assumption is invalid for the input, one can handle the case of $\ell(OPT) = 0$ by simply dismissing all the elements whose linear cost is positive and then using an algorithm for unregulated submodular maximization on the remaining elements.

makes Algorithm 1 return a solution S with the guarantee

$$\frac{g(S) - \ell(S)}{g(OPT) - \ell(OPT)} \geq \frac{1 + \beta_{OPT} - \sqrt{1 + 2\beta_{OPT}}}{2\beta_{OPT}} - \varepsilon' .$$

In order to get the strongest guarantee using Corollary 6, we need access to the set OPT (and consequently β_{OPT} and r_{OPT}); but, unfortunately, none of these is known a priori. It turns out, however, that by trying a relatively small number of guesses for β_{OPT} and r_{OPT}, one can get a guarantee that is almost as good as the one that can be obtained by knowing these exact values. The full version of our proposed algorithm, named DISTORTED-STREAMING, is based on this idea. Its pseudocode is given as Algorithm 2, and assumes that $\delta > 0$ is an accuracy parameter.

Algorithm 2: DISTORTED-STREAMING

```
1 $\Lambda \leftarrow \{\varepsilon(1+\delta)^i \mid 0 \leq i \leq \lfloor \log_{1+\delta}(1/(2\varepsilon)) \rfloor\}$.
2 for every $\zeta \in \Lambda$ in parallel do
3     Calculate $\beta \leftarrow \frac{4\zeta}{(1-2\zeta)^2}$.
4     Calculate $r \leftarrow \frac{\beta}{2\sqrt{1+2\beta}}$.  // This is the formula from Corollary 6.
5     Run THRESHOLD-STREAMING (Algorithm 1) with $r$.
6 return the best among the solutions found by all the copies of THRESHOLD-STREAMING executed.
```

One can see that the value of r passed to every copy of THRESHOLD-STREAMING by DISTORTED-STREAMING is at least $\varepsilon/3$, and thus, the number of elements kept by each such copy is at most $O(\varepsilon^{-1}(\log k + \log \varepsilon^{-1}))$. Furthermore, the number of elements kept by DISTORTED-STREAMING is larger than that by a factor of at most $1 + \log_{1+\delta}(1/(2\varepsilon)) = O(\delta^{-1} \cdot \log(\varepsilon^{-1}))$. Theorem 1, given in Section 1, studies the approximation guarantee of DISTORTED-STREAMING.

4. Distributed Algorithm

The exponential growth of data makes it difficult to process, or even store, the entire data on a single machine. For this reason, there is an urgent need to develop distributed or parallel computing methods to process massive datasets. Distributed algorithms in the Map-Reduce model have shown promising results in several problems related to submodular maximization (Mirzasoleiman et al., 2013; Mirrokni & Zadimoghaddam, 2015; Barbosa et al., 2015; 2016). In this section, we present a distributed solution for Problem (1), named DISTORTED-DISTRIBUTED, which appears as Algorithm 3. Our algorithm uses DISTORTED-GREEDY proposed by Harshaw et al. (2019) as a subroutine.

Our distributed solution is based on a framework suggested by Barbosa et al. (2016) for converting greedy-like sequential algorithms into distributed ones. However, its analysis is a generalization of ideas from (Barbosa et al., 2015) rather than being a direct adaptation of the analysis given by (Barbosa et al., 2016). This allows us to get an algorithm which uses asymptotically the same number of computational rounds as the algorithm of (Barbosa et al., 2016), but does not need to keep multiple copies of the data as is done by the last algorithm. We would also like to point out that Barbosa et al. (2016) have proposed a variant of their algorithm that avoids data replication, but it does so at the cost of increasing the number of rounds from $\Theta(1/\varepsilon)$ to $\Theta(1/\varepsilon^2)$.

DISTORTED-DISTRIBUTED is a distributed algorithm within the Map-Reduce framework using $\lceil 1/\varepsilon \rceil$ rounds of computation, where $\varepsilon \in (0, 1/2]$ is a parameter controlling the quality of the algorithm's output. In the first round of computation, DISTORTED-DISTRIBUTED distributes the elements among m machines by independently sending each element $u \in \mathcal{N}$ to a uniformly random machine. Each machine i then runs DISTORTED-GREEDY on its data and forwards the resulting solution $S_{1,i}$ to all other machines (in general, we denote by $S_{r,i}$ the solution calculated by machine i in round r). The next rounds repeat this operation, except that the data of each machine now includes also the elements included in any solution calculated (by any machine) during the previous rounds. At the end of the last round, machine number 1 outputs the final solution, which is the best solution among the solution computed by this machine in the last round and the solutions computed by all the machines in the previous rounds. Theorem 2 analyzes the approximation guarantee of DISTORTED-DISTRIBUTED.

Algorithm 3: DISTORTED-DISTRIBUTED

```
1 for $r = 1$ to $\lceil \varepsilon^{-1} \rceil$ do
2     for each $u \in \mathcal{N}$ do
3         Assign element $u$ to a machine chosen uniformly at random (and independently) among $m$ machines.
4         Let $\mathcal{N}_{r,i}$ denote the elements assigned to machine $i$ in this round.
5     for $i = 1$ to $m$ do
6         Run DISTORTED-GREEDY on the set $\mathcal{N}_{r,i} \cup (\cup_{r'=1}^{r-1} \cup_{i'=1}^{m} S_{r',i'})$ to get the solution $S_{r,i}$ of size at most $k$ .
7     if $r < \varepsilon^{-1}$ then Forward the solutions $S_{r,i}$, for every integer $1 \leq i \leq m$, to all the machines.
8     else return a set $D$ maximizing $g(D) - \ell(D)$ among all sets in $\{S_{r,1}\} \cup \{S_{r',i'} \mid 1 \leq r' < r$ and $1 \leq i' \leq m\}$.
```

5. Mode Finding of SLC Distributions

In Section 1, we discussed the power of sampling from discrete probabilistic models (specifically, strongly log-concave distributions), which encourage negative correlation, for data summarization. In this regard, recently, Robinson et al. (2019) established a notion of γ-additively weak submodularity for strongly log-concave (SLC) functions. Using this newly defined property, we can apply our algorithms to obtain a performance guarantee for mode finding of a class of distributions derived from SLC. Following is the definition of γ-additively weak submodular functions.

Definition 7 (Definition 1, (Robinson et al., 2019))**.** *A set function $\rho\colon 2^{\mathcal{N}} \to \mathbb{R}$ is γ-additively weak submodular if for any $S \subseteq \mathcal{N}$ and $u, v \in \mathcal{N} \setminus S$ with $u \neq v$, we have*

$$\rho(S) + \rho(S \cup \{u, v\}) \leq \gamma + \rho(S \cup \{u\}) + \rho(S \cup \{v\}) \enspace .$$

In order to maximize a γ-additively weak submodular function ρ, we show that, with a little modification, ρ can be converted to a submodular function Λ (defined in Lemma 8). We then show that Λ, in its turn, can be rewritten as the difference between a non-negative monotone submodular function and a modular function (Lemma 9), which allows us to optimize Λ using the results of Theorem 4 and Theorem 2. Finally, we show that even though we use these algorithms to optimize Λ, the solution they provide has a good guarantee with respect to the original γ-additively weak submodular function ρ. With this new formulation, we improve the theoretical guarantees of Robinson et al. (2019) in the offline setting and provide streaming and distributed solutions for the mode finding problem under a cardinality constraint. Specifically, by using either our proposed streaming or distributed algorithms (depending on the setting), we can get a scalable solution with a guarantee with respect to ρ, and in particular, a guarantee for the task of finding the mode of an SLC distribution. We should also point out that the distorted greedy algorithm (Harshaw et al., 2019, Algorithm 1) can be used in a similar way to optimize ρ in the offline setting.

Lemma 8. *For a γ-additively weak submodular function ρ, the function $\Lambda(S) \triangleq \rho(S) - \frac{\gamma}{2} \cdot |S| \cdot (|S| - 1)$ is submodular.*

Now, let us define the modular function $\ell(S) = \sum_{u \in S} \ell_u$, where $\ell_u \triangleq \max\{\Lambda(\mathcal{N} \setminus u) - \Lambda(\mathcal{N}), 0\} = \max\{\rho(\mathcal{N} \setminus u) - \rho(\mathcal{N}) + \gamma \cdot (|\mathcal{N}| - 1), 0\}$.

Lemma 9. *The function $g(S) \triangleq \Lambda(S) + \ell(S)$ is monotone and submodular. Furthermore, if $\rho(\varnothing) \geq 0$, then $g(S)$ is also non-negative because $\ell(\varnothing) = 0$.*

As promised, we now show that by optimizing Λ under a cardinality constraint using either of our proposed algorithms yields a scalable solution with a guarantee with respect to ρ.

Corollary 10. *Assume $\rho\colon 2^{\mathcal{N}} \to \mathbb{R}$ is a γ-additively weak submodular function obeying $\rho(\varnothing) \geq 0$. Then, when given Λ as the objective function,* DISTORTED-DISTRIBUTED *(Algorithm 3) returns a solution R such that*

$$\mathbb{E}[\rho(R)] \geq (1 - \varepsilon) \left[(1 - e^{-1})\rho(\texttt{OPT}) - e^{-1} \cdot \ell(\texttt{OPT})\right] - \frac{\gamma \cdot [(1 - e^{-1}) \cdot l(l - 1) - \mathbb{E}[|R|(|R| - 1)]]}{2} \enspace ,$$

where $\texttt{OPT} \in \arg\max_{|S| \leq k} \rho(S)$ and $l = |\texttt{OPT}| \leq k$.

The following corollary shows the guarantee obtained by THRESHOLD-STREAMING as a function of the input parameter r. When the best choice for r is unknown, DISTORTED-STREAMING roughly obtains this guarantee for the best value of r, as discussed in Section 3.

Corollary 11. *Assume $\rho\colon 2^{\mathcal{N}} \to \mathbb{R}$ is a γ-additively weak submodular function obeying $\rho(\varnothing) \geq 0$. Then, when given Λ as the objective function,* THRESHOLD-STREAMING *(Algorithm 1) returns a solution R such that $\rho(R)$ is at least*

$$(h(r) - \varepsilon) \cdot \rho(\texttt{OPT}) - (\alpha(r) - r - 1 + \varepsilon) \cdot \ell(\texttt{OPT}) - \frac{\gamma \cdot [(h(r) - \varepsilon) \cdot l \cdot (l - 1) - |R| \cdot (|R| - 1)]}{2} \enspace ,$$

where $\texttt{OPT} \in \arg\max_{|S| \leq k} \rho(S)$ and $l = |\texttt{OPT}| \leq k$.

Note that in (Robinson et al., 2019, Theorem 12) and Corollary 10, if the value of the linear function is considerably larger than the values of functions η or ρ, then the bounds given by these results can be negative (and thus, trivial). The main explanation for this phenomenon is that the distorted greedy algorithm does not take into account the relative importance of g and ℓ to the value of the optimal solution. On the other hand, the **distinguishing feature** of our streaming algorithm is that, by guessing the value of $\beta_{\texttt{OPT}}$, it can assign weights to the importance of the submodular and modular terms in the best possible way. Therefore, DISTORTED-STREAMING, even in the scenarios where the linear cost is large, can find solutions with a non-trivial provable guarantee. In the experiments presented in Appendix D.2, we showcase that: (i) DISTORTED-STREAMING could be used for mode finding of strongly log-concave distributions with a provable guarantee, and (ii) choosing an accurate estimation of $\beta_{\texttt{OPT}}$ plays an important role in this optimization procedure. In Appendix D.2, we compare the performance of DISTORTED-STREAMING with several other algorithms on the problem of mode finding for an SLC distribution.

6. Experiments

In this section, we present the experimental studies we have performed to show the applicability of our approach. In the first set of experiments (Section 6.1), we compare the performance of our proposed streaming algorithm with that of DISTORTED-GREEDY (Harshaw et al., 2019), vanilla greedy, and sieve-streaming (Badanidiyuru et al., 2014). The main message of the first experiments is that our proposed distorted-streaming algorithm outperforms both vanilla greedy and sieve streaming, and performs comparably with respect to the distorted-greedy algorithm in terms of the objective value—despite the fact that our proposed algorithm makes only a single pass over the data, while distorted-greedy has to make k passes (which can be as large as $\Theta(n)$ in the worst case).[6]

In the second set of experiments (Section 6.2), we compare DISTORTED-DISTRIBUTED with distributed greedy. In the final experiments (Section 6.3), we demonstrate the power of our proposed regularized model by comparing it with the alternative approach of maximizing a submodular function subject to cardinality and single knapsack constraints. In the latter case, the goal of the knapsack constraint is to limit the linear function ℓ to a pre-specified budget while the algorithm tries to maximize the submodular function g. For supplementary experimental evaluations refer to Appendices D.2 and E.

6.1. How Effective is DISTORTED-STREAMING?

In this experiment, we compare DISTORTED-STREAMING with distorted-greedy, greedy, and sieve-streaming in the setting studied in (Harshaw et al., 2019, Section 5.2). In this setting, there is a submodular function f over the vertices of a directed graph $G = (V, E)$. To define this function, we first need to have a weight function $w\colon V \to \mathbb{R}_{\geq 0}$ on the vertices. For a given vertex set $S \subseteq V$, let $N(S)$ denote the set of vertices which are pointed to by S, i.e., $N(S) \triangleq \{v \in V \mid \exists u \in S \text{ such that } (u, v) \in E\}$. Then, we have $f(S) \triangleq g(S) - \ell(S) = \sum_{u \in N(S) \cup S} w_u - \sum_{u \in S} \ell_u$. Following Harshaw et al. (2019), we assigned a weight of 1 to all nodes and set $\ell_u = 1 + \max\{0, d_u - q\}$, where d_u is the out-degree of node u in the graph $G(V, E)$ and q is a parameter. In our experiment, we used real-world graphs from (Leskovec & Krevl, 2014), set $q = 6$, and ran the algorithms for varying cardinality constraint k. In Fig. 2, we observe that for all four networks, distorted greedy, which is an offline algorithm, achieves the highest objective values. Furthermore, we observe that DISTORTED-STREAMING consistently outperforms both greedy and sieve-streaming, which demonstrates the effectiveness of our proposed method.

[6]We note that some algorithms from the literature improve over vanilla greedy in terms of speed (Stochastic greedy (Mirzasoleiman et al., 2015) and FAST (Breuer et al., 2019)) or over sieve streaming in terms of the memory complexity (Sieve-streaming++ (Kazemi et al., 2019)). However, in all these cases the improved algorithm is no better than the original algorithm in terms of the approximation quality (which is the main purpose of our experiments), and therefore, it suffices for us to compare against the algorithms that we have listed.

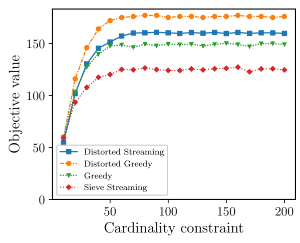
In Section 1, we discussed how shifting the function $f \triangleq g - \ell$ by a constant value of C and making it non-negative would affect the approximation factors of the existing algorithms (including vanilla greedy) for maximizing non-negative submodular functions. We showed that different values of C result in different theoretical approximation guarantees from running vanilla greedy over the function $f + C$. On the other hand, shifting a function by a constant value does not change the marginal gains of the elements. Therefore, the solution returned by the vanilla greedy is the same for all different values of C. As a result, the improvement we gain over the vanilla greedy in this experiment confirms the need for an approach (from both theoretical and practical perspectives) beyond applying existing algorithms for non-negative submodular functions to shifted versions of f.

6.2. Distributed Setting

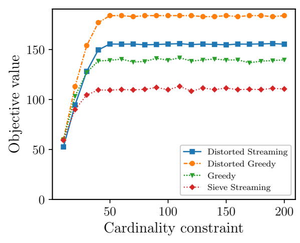
In this section, we compare DISTORTED-DISTRIBUTED with the distributed greedy algorithm of Barbosa et al. (2016). We evaluate the performance of these algorithms over several large graphs (Leskovec & Krevl, 2014) in the setting of Section 6.1 under a cardinality constraint $k = 1{,}000$, where we set $q = 50$. For both algorithms, we set the number of computational rounds to 10. The first graph is the Amazon product co-purchasing network with $n = 334{,}863$ vertices; the second one is the DBLP collaboration network with $n = 317{,}080$ vertices; the third graph is Youtube with $n = 1{,}134{,}890$ vertices; and the last graph we consider is the Pokec social network, the most popular online social network in Slovakia, with $n = 1{,}632{,}803$ vertices. For each graph, we set the number of machines to $m = \lceil n/4000 \rceil$. In Fig. 3, we can see that the objective values of DISTORTED-DISTRIBUTED exceed the results of distributed greedy for all four graphs.

6.3. Regularized Data Summarization

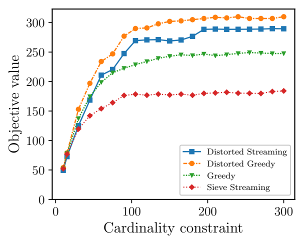
In this section and Appendix E, through an extensive set of experiments, we answer the following two questions: (i) How does DISTORTED-STREAMING perform with respect to sieve-streaming and distorted greedy on real-world data summarization tasks? (ii) Is our proposed modeling approach (maximizing diversity while considering costs of items as a regularization term) favorable to methods which try to maximize a submodular function subject to a knapsack constraint? To do this, we consider three state-of-the-art algorithms for solving the problem of submodular maximization with a cardinality and a knapsack constraint: FANTOM (Mirzasoleiman et al., 2016a), Fast (Badanidiyuru & Vondrák, 2014) and Vanilla Greedy Dynamic Program (Mizrachi et al., 2019). For the sake of fairness of experiments, we used these three algorithms to maximize the submodular function g under 50 different knapsack capaci-



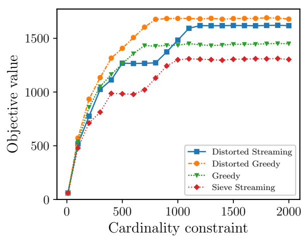
(a) Social graph



(b) Facebook ego network



(c) EU Email



(d) Wikipedia vote

Figure 2. Directed vertex cover: comparison of objective values, with a varying cardinality constraint k, for four different real-world networks.

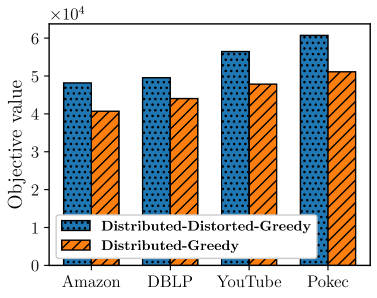


Figure 3. Comparison of DISTORTED-DISTRIBUTED with the distributed greedy algorithm subject to a cardinality constraint $k = 1{,}000$. The number of computational rounds is set to 10.

ties c in the interval $0.1 \leq c \leq 100$ and reported the solution maximizing $g(S) - \ell(S)$. For the computational complexities of these algorithms, we report the number of oracle calls used by a single one out of their 50 runs (even though we report the best output in any of these runs), giving these offline algorithms a considerable edge in the comparison.

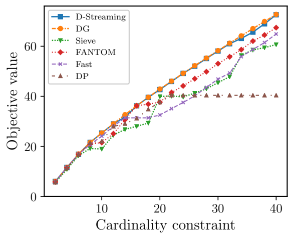
We consider in this section an online video summarization application, where a stream of video frames comes, and one needs to provide a set of representative frames as the summary of the whole video. More formally, the objective is to select a subset of frames in order to maximize a utility function $g(S)$ (representing diversity), while minimizing a non-negative modular function $\ell(S)$ capturing the total entropy of the selection (the entropy of the set S could be interpreted as a proxy of the storage size of S).

We used the pre-trained ResNet-18 model (He et al., 2016) to extract features from frames of each video. Then, given a set of frames, we defined a matrix M such that $M_{ij} = e^{-\text{dist}(x_i,x_j)}$, where $\text{dist}(x_i, x_j)$ denotes the distance between the feature vectors of the i-th and j-th frames, respectively. One can think of M as a similarity matrix among different frames of a video. The utility of a set $S \subseteq V$ is defined as a non-negative monotone submodular function $g(S) = \log\det(\mathbf{I} + \alpha M_S)$, where $\mathbf{I}$ is the identity matrix, α is a positive scalar and M_S is the principal sub-matrix of the similarity matrix M indexed by S. Informally, this function measures the diversity of the vectors in S. To sum-up, we want to maximize the following function under a cardinality constraint k: $f(S) \triangleq g(S) - \ell(S) = \log\det(\mathbf{I} + \alpha M_S) - \sum_{u \in S} \mathrm{H}_u$, where H_u is the entropy of frame u.

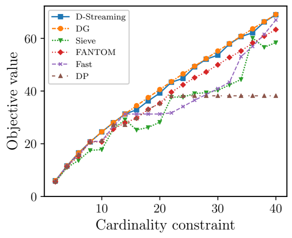
In the first experiment, we summarized the frames of videos 13 and 15 from the VSUMM dataset (De Avila et al., 2011),[7] and compared the above-mentioned algorithms based on their objective values and number of oracle values for varying cardinality constraint k. From Figs. 4a and 4b we conclude that (i) the quality of the solutions returned by DISTORTED-STREAMING is as good as the quality of the results of distorted greedy, (ii) distorted greedy clearly outperforms sieve-streaming, and (iii) the objective values of DISTORTED-STREAMING and distorted greedy are both larger than the corresponding values produced by Greedy Dynamic Program, Fast and FANTOM. This confirms that directly maximizing the function f provides higher utilities versus maximizing the function g and setting a knapsack constraint over the modular function ℓ. In Figs. 4c and 4d, we observe that the computational complexity of DISTORTED-STREAMING and sieve streaming is several orders of magnitudes better than the computation complexity of the other algorithms, which is consistent with their need to make only a single pass over the data.

Next, we study the effect of the linear cost function (in other words, the importance we give to the entropy of frames) on the set of selected frames. For this reason, we ran DISTORTED-STREAMING on the frames from video number 14. The objective function is $f(S) \triangleq g(S) - \lambda \cdot \ell(S)$ for $\lambda \in \{0, 0.5, 1.0\}$. In this experiment, we set the cardinality constraint to $k = 6$. In Fig. 4e, we observe that by increasing λ the entropy of the selected frames decreases. This is evident from the fact that the color diversity of pixels in each frame reduces for larger values of λ. Consequently, the representativeness of the selected subset decreases. In-

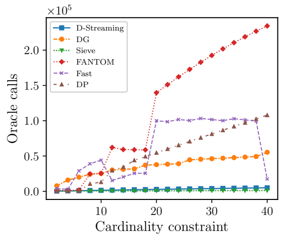
[7] `https://sites.google.com/site/vsummsite/`



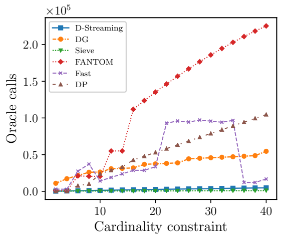
(a) Video number 13



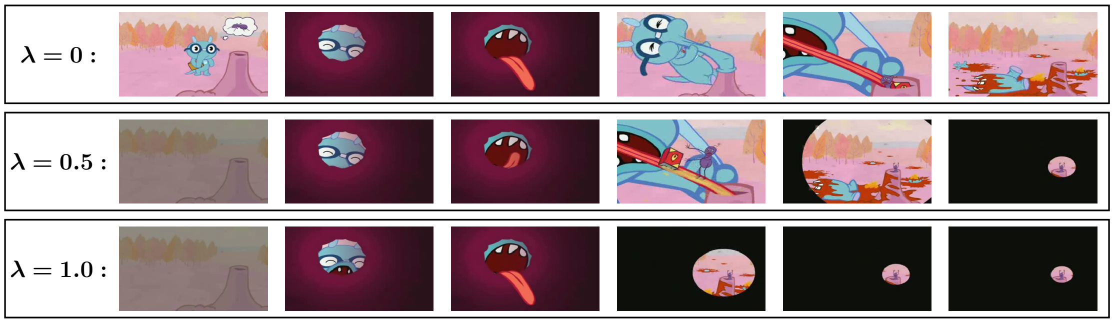
(b) Video number 15



(c) Video number 13



(d) Video number 13



(e) The summaries produced by DISTORTED-STREAMING for video number 14.

Figure 4. Movie frame summarization: For each frame u the linear cost ℓ_u is the entropy of that frame. (a) and (b) compare the objective values. (c) and (d) compare the computational complexities based on the number of oracle calls. In experiment (e), the input function is $f(S) \triangleq g(S) - \lambda \cdot \ell(S)$ for $\lambda \in \{0, 0.5, 1.0\}$, where we set the cardinality constraint to $k = 6$.

deed, while it is easy to understand the whole story of this animation from the output produced for $\lambda = 0$, some parts of the story are definitely missing if we set λ to 1.0.

7. Conclusion

In this paper, we proposed scalable methods for maximizing a *regularized* submodular function expressed as the difference between a non-negative monotone submodular function g and a modular function ℓ. We developed the first one-pass streaming algorithm for maximizing a regularized submodular function subject to a cardinality constraint with a theoretical performance guarantee, and also presented a distributed algorithm matching the guarantees of state-of-the-art offline algorithms using only $O(1/\varepsilon)$ rounds of MapReduce computation. Moreover, even for the unregularized case, our distributed algorithm improves the memory and communication complexity of the existing work by a factor of $\Theta(1/\varepsilon)$. We also empirically studied the performance of our scalable methods on a set of real-world applications.

Acknowledgements

Amin Karbasi is partially supported by NSF (IIS-1845032), ONR (N00014-19-1-2406), AFOSR (FA9550-18-1-0160), and TATA Sons Private Limited.

References

Alaluf, N., Ene, A., Feldman, M., Nguyen, H. L., and Suh, A. Optimal streaming algorithms for submodular maximization with cardinality constraints. *CoRR*, abs/1911.12959, 2020.

Badanidiyuru, A. and Vondrák, J. Fast algorithms for maximizing submodular functions. In *SODA*, pp. 1497–1514, 2014.

Badanidiyuru, A., Mirzasoleiman, B., Karbasi, A., and Krause, A. Streaming Submodular Maximization:Massive Data Summarization on the Fly. In *KDD*, pp. 671–680, 2014.

Barbosa, R. d. P., Ene, A., Nguyen, H., and Ward, J. The power of randomization: Distributed submodular maximization on massive datasets. In *ICML*, pp. 1236–1244, 2015.

Barbosa, R. d. P., Ene, A., Nguyen, H. L., and Ward, J. A New Framework for Distributed Submodular Maximization. In *FOCS*, pp. 645–654, 2016.

Borcea, J., Brändén, P., and Liggett, T. Negative dependence and the geometry of polynomials. *Journal of the American Mathematical Society*, 22(2):521–567, 2009.

Breuer, A., Balkanski, E., and Singer, Y. The FAST Algorithm for Submodular Maximization. *CoRR*, abs/1907.06173, 2019.

Buchbinder, N. and Feldman, M. Submodular Functions Maximization Problems. In *Handbook of Approximation Algorithms and Metaheuristics*, pp. 771–806. Chapman and Hall/CRC, 2018.

Buchbinder, N., Feldman, M., and Schwartz, R. Online submodular maximization with preemption. In *SODA*, pp. 1202–1216, 2015.

Chekuri, C., Gupta, S., and Quanrud, K. Streaming algorithms for submodular function maximization. In *International Colloquium on Automata, Languages, and Programming*, pp. 318–330. Springer, 2015.

Das, A. and Kempe, D. Submodular meets spectral: greedy algorithms for subset selection, sparse approximation and dictionary selection. In *ICML*, pp. 1057–1064, 2011.

De Avila, S. E. F., Lopes, A. P. B., da Luz Jr, A., and de Albuquerque Araújo, A. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56–68, 2011.

Djolonga, J. and Krause, A. From map to marginals: Variational inference in bayesian submodular models. In *NeurIPS*, pp. 244–252, 2014.

Elenberg, E. R., Dimakis, A. G., Feldman, M., and Karbasi, A. Streaming Weak Submodularity: Interpreting Neural Networks on the Fly. In *NeurIPS*, pp. 4047–4057, 2017.

Elenberg, E. R., Khanna, R., Dimakis, A. G., Negahban, S., et al. Restricted strong convexity implies weak submodularity. *The Annals of Statistics*, 46(6B):3539–3568, 2018.

Ene, A., Nikolakaki, S. M., and Terzi, E. Team formation: Striking a balance between coverage and cost. *CoRR*, abs/2002.07782, 2020. URL https://arxiv.org/abs/2002.07782.

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

Feldman, M. Guess free maximization of submodular and linear sums. *Algorithmica*, 83(3):853–878, Mar 2021.

Feldman, M., Naor, J., Schwartz, R., and Ward, J. Improved approximations for k-exchange systems - (extended abstract). In *ESA*, pp. 784–798, 2011.

Feldman, M., Harshaw, C., and Karbasi, A. Greed is good: Near-optimal submodular maximization via greedy optimization. In *COLT*, pp. 758–784, 2017.

Feldman, M., Karbasi, A., and Kazemi, E. Do Less, Get More: Streaming Submodular Maximization with Subsampling. In *NeurIPS*, pp. 730–740, 2018.

Feldman, M., Norouzi-Fard, A., Svensson, O., and Zenklusen, R. The One-way Communication Complexity of Submodular Maximization with Applications to Streaming and Robustness. *arXiv preprint arXiv:2003.13459*, 2020.

Frieze, A. M. A cost function property for plant location problems. *Mathematical Programming*, 7(1):245–248, 1974.

Gotovos, A. Strong log-concavity does not imply log-submodularity. *arXiv preprint arXiv:1910.11544*, 2019.

Gotovos, A., Hassani, H., and Krause, A. Sampling from probabilistic submodular models. In *NeurIPS*, pp. 1945–1953, 2015.

Gurvits, L. A polynomial-time algorithm to approximate the mixed volume within a simply exponential factor. *Discrete & Computational Geometry*, 41(4):533–555, 2009.

Gygli, M., Grabner, H., and Van Gool, L. Video summarization by learning submodular mixtures of objectives. In *CVPR*, pp. 3090–3098, 2015.

Haba, R., Kazemi, E., Feldman, M., and Karbasi, A. Streaming submodular maximization under a k-set system constraint. In *International Conference on Machine Learning*, pp. 3939–3949. PMLR, 2020.

Harper, F. M. and Konstan, J. A. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

Harshaw, C., Feldman, M., Ward, J., and Karbasi, A. Submodular Maximization beyond Non-negativity: Guarantees, Fast Algorithms, and Applications. In *ICML*, pp. 2634–2643, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *CVPR*, pp. 770–778, 2016.

Iyer, R. and Bilmes, J. Submodular point processes with applications to machine learning. In *Artificial Intelligence and Statistics*, pp. 388–397, 2015.

Iyer, R. K. and Bilmes, J. A. Submodular optimization with submodular cover and submodular knapsack constraints. *Advances in neural information processing systems*, 26: 2436–2444, 2013.

Kazemi, E., Zadimoghaddam, M., and Karbasi, A. Scalable Deletion-Robust Submodular Maximization: Data Summarization with Privacy and Fairness Constraints. In *ICML*, pp. 2549–2558, 2018.

Kazemi, E., Mitrovic, M., Zadimoghaddam, M., Lattanzi, S., and Karbasi, A. Submodular Streaming in All Its Glory: Tight Approximation, Minimum Memory and Low Adaptive Complexity. In *ICML*, pp. 3311–3320, 2019.

Krause, A. and Golovin, D. Submodular Function Maximization. In *Tractability: Practical Approaches to Hard Problems*. Cambridge University Press, 2012.

Kulesza, A. and Taskar, B. Determinantal Point Processes for Machine Learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

Lee, J., Mirrokni, V. S., Nagarajan, V., and Sviridenko, M. Maximizing nonmonotone submodular functions under matroid or knapsack constraints. *SIAM J. Discrete Math.*, 23(4):2053–2078, 2010.

Leskovec, J. and Krevl, A. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data, June 2014.

Lin, H. and Bilmes, J. A. A Class of Submodular Functions for Document Summarization. In *HLT*, pp. 510–520, 2011.

Lindgren, E. M., Wu, S., and Dimakis, A. G. Sparse and greedy: Sparsifying submodular facility location problems. In *NeurIPS Workshop on Optimization for Machine Learning*, 2015.

Liu, P. and Vondrák, J. Submodular optimization in the mapreduce model. In *SOSA*, pp. 18:1–18:10, 2019.

Lovász, L. Submodular functions and convexity. In *Mathematical Programming The State of the Art*, pp. 235–257. Springer, 1983.

Mirrokni, V. and Zadimoghaddam, M. Randomized composable core-sets for distributed submodular maximization. In *STOC*, pp. 153–162, 2015.

Mirzasoleiman, B., Karbasi, A., Sarkar, R., and Krause, A. Distributed submodular maximization: Identifying representative elements in massive data. In *NeurIPS*, pp. 2049–2057, 2013.

Mirzasoleiman, B., Badanidiyuru, A., Karbasi, A., Vondrak, J., and Krause, A. Lazier than Lazy Greedy. In *AAAI Conference on Artificial Intelligence*, pp. 1812–1818, 2015.

Mirzasoleiman, B., Badanidiyuru, A., and Karbasi, A. Fast Constrained Submodular Maximization: Personalized Data Summarization. In *ICML*, pp. 1358–1367, 2016a.

Mirzasoleiman, B., Karbasi, A., Sarkar, R., and Krause, A. Distributed Submodular Maximization. *Journal of Machine Learning Research (JMLR)*, 17:1–44, 2016b.

Mitrovic, M., Kazemi, E., Zadimoghaddam, M., and Karbasi, A. Data Summarization at Scale: A Two-Stage Submodular Approach. In *ICML*, pp. 3593–3602, 2018.

Mizrachi, E., Schwartz, R., Spoerhase, J., and Uniyal, S. A Tight Approximation for Submodular Maximization with Mixed Packing and Covering Constraints. In *International Colloquium on Automata, Languages, and Programming, (ICALP)*, pp. 85:1–85:15, 2019.

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.

Robinson, J., Sra, S., and Jegelka, S. Flexible Modeling of Diversity with Strongly Log-Concave Distributions. In *NeurIPS*, pp. 15199–15209, 2019.

Sviridenko, M., Vondrák, J., and Ward, J. Optimal approximation for submodular and supermodular optimization with bounded curvature. *Math. Oper. Res.*, 42(4):1197–1218, 2017.

Tschiatschek, S., Iyer, R. K., Wei, H., and Bilmes, J. A. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *NeurIPS*, pp. 1413–1421, 2014.

Wei, K., Iyer, R., and Bilmes, J. Submodularity in data subset selection and active learning. In *ICML*, pp. 1954–1963, 2015.

Yelp. Yelp Academic Dataset. https://www.kaggle.com/yelp-dataset/yelp-dataset, 2019a.

Yelp. Yelp Dataset. https://www.yelp.com/dataset, 2019b.