Big Data Summarization: The Role of Submodularity
Amin Karbasi
Yale
From Big Data to Small Data

- Summarization through the lens of submodularity
- Efficient algorithms for data summarization
Summarization

Key challenge:

Extract small, representative subset out of a massive data set
Summarization: Benefits

- Remove redundancy
- Run the algorithms efficiently AGAIN
- Obtain more accurate results

Diminishing Returns

Diversity (Essential information)

Coverage (No loss of information)
Balls in Urns (Toy Example)

Utility: number of distinct colors

Initial Value: 2
New Value: 3

Initial Value: 3
New Value: 3
Submodularity

- Diminishing returns property for set functions.

\[ V = \left\{ \begin{array}{c} \text{Rose} \end{array} \right\} \]

\[ f \left( \left\{ \begin{array}{c} \text{Rose} \end{array} \right\} \right) - f \left( \left\{ \begin{array}{c} \text{Rose} \end{array} \right\} \right) \geq \]

\[ f \left( \left\{ \begin{array}{c} \text{Rose} \end{array} \right\} \right) - f \left( \left\{ \begin{array}{c} \text{Rose} \end{array} \right\} \right) \]

\[ \forall A \subseteq B \subseteq V \text{ and } x \notin \]

\[ f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) \]

\[ \forall A \subseteq B \]

\[ f(B) \geq f(A) \]
Image Summarization

- Given a huge set of images, summarise it to a few exemplars.
Example: Exemplar Based Clustering

\[ L(A) = \frac{1}{V} \sum_{s \in V} \min_{c \in A} d(x_s, x_c) \]

\[ f(S) = L(\{e_0\}) - L(S \cup \{e_0\}) \]
Example: Nonparametric learning in massive data sets

\[ f(x) \]

- \[ f_V(x) = \sum_{x' \in V} \alpha_x k(x', x) \]
- \[ S \subseteq V, |S| \ll |V| \]
- \[ f_S(x) = \sum_{x' \in S} \hat{\alpha}_x k(x', x) \]
Example: Nonparametric learning in massive data sets

Pick active set to maximally capture variation.

\[ F_H(A) = \log |I + K_{AA}| \]

\[ K_{AA} = \begin{pmatrix} k(x_1, x_1) & \cdots & k(x_1, x_{|A|}) \\ \vdots & \ddots & \vdots \\ k(x_{|A|}, x_1) & \cdots & k(x_{|A|}, x_{|A|}) \end{pmatrix} \]
Selecting representative elements via submodular optimization

- Wish to select small, representative subset out of large data set
  - Clustering [NJB ‘05, GK’10]
  - Recommendation [LKGVVVG ’06, KSG’11, YG’12]
  - Document summarization [LB’11, LB’12]
  - Scaling up non-parametric learning [S’04, GK’10]
  - Corpus subset selection [LB’11]
  - Compostable core-sets [IMMM’14, MZ’15]

- Often, natural utility functions are monotone submodular

\[ A^* = \arg \max_{|A| \leq k} F(A) \]
Given a monotone submodular function $F$, and a cardinality constraint $K$, find a set $A^*$ such that

$$A^* = \arg\max_{|A| \leq k} F(A)$$

Centralized Solution

Streaming Solution

Distributed Solution
Centralized Solution

Data can be loaded on the main memory
The Greedy Algorithm

Problem: Find $S^* = \arg\max \{F(S) : |S| \leq k\}$

\[ \Delta f = \{ \ldots \} \]

Greedy algorithm:

Start with $A = \emptyset$

For $i = 1$ to $j$,

\[ A = A \cup \{ s^* \} \]

\[ F(S_{\text{greedy}}) \geq (1 - \frac{1}{e}) F(S^*) \]

Theorem [Nemhauser, Wolsey & Fisher ’78]

For monotonic submodular functions, Greedy algorithm gives constant factor approximation.
Problem: Greedy can be impractical

- For a ground set $V$ of size $n$, standard greedy algorithm needs $O(n \cdot k)$ function evaluations to find a summary of size $k$.

- In many data intensive applications, evaluating $f$ is expensive and running the standard greedy algorithm is infeasible.

Is it possible to have an algorithm that “does not depend on $k$” at all and scales linearly with the data size $n$?
SubSample-Greedy Algorithm

Problem: Find $S^* = \arg\max \{F(S) : |S| \leq k\}$

Sample $n/k$ points per iteration

Subsample Greedy

For monotonic submodular functions, Random Greedy gives constant factor approximation using only $n$ function evaluation

$$E[f(S_{\text{rand greedy}})] \geq (1-1/e)^2 f(S^*)$$

"Differentially Private Submodular Maximization", ICML’17
Mitrovic, Bun, Krause, Karbasi
Random-Greedy Algorithm

Problem: Find $S^* = \text{argmax} \ \{F(S) : |S| \leq k\}$

**Random Greedy**

For monotonic submodular functions, Random Greedy gives constant factor approximation using only $n$ function evaluation

$$E[f(S_{\text{rand greedy}})] \geq (1-1/e-\varepsilon)f(S^*)$$

Sample $n/k \log(1/\varepsilon)$ points per iteration
**Algorithm** Rand-Greedy

**Input:** \( f : 2^V \rightarrow \mathbb{IR}_+ , k \in \{1, \ldots , n\} \).

**Output:** A set \( A \subseteq V \) satisfying \(|A| \leq k\).

1: \( A \leftarrow \emptyset \).
2: for \((i \leftarrow 1; i \leq k; i \leftarrow i + 1)\) do
3: \( R \leftarrow \) a random subset obtained by sampling \( s_i \) random elements from \( V \setminus A \).
4: \( a_i = \text{argmax}_{a \in R} (a \mid A) \).
5: \( A \leftarrow A \cup \{a_i\} \).
6: return \( A \).

- First centralized algorithm for cardinality-constrained submodular maximization with
  - Constant factor **approximation** guarantee
    \[ E(f(S)) \geq (1-1/e-\epsilon) \text{OPT} \]
  - **Time** is linear in the size of the data and
  - is independent of the cardinality constraint, i.e.
    \[ O(n \log 1/\epsilon) \] function evaluations
  - Assuming nothing but monotone submodularity

"Lazier Than Lazy Greedy", AAAI’15
(Mirzasoleiman, Badanidiyuru, Karbasi, Vondrak, Krause)
$F_H(A) = \log |I + K_{AA}|$

$K_{AA} = \begin{pmatrix} k(x_1, x_1) & \ldots & k(x_1, x_{|A|}) \\ \vdots & \ddots & \vdots \\ k(x_{|A|}, x_1) & \ldots & k(x_{|A|}, x_{|A|}) \end{pmatrix}$

$L(A) = \frac{1}{V} \sum_{s \in V} \min_{c \in A} d(x_s, x_c)$

$f(S) = L(\{e_0\}) - L(S \cup \{e_0\})$
Parkinson dataset consists of 5,875 data points with 22 attributes.

Classic Greedy is too expensive.
Benchmarks:

- **Standard greedy**: the output is the $k$ data points selected by the greedy algorithm. This algorithm is not applicable on large datasets.

- **Lazy greedy**: the output produced by the accelerated greedy method [M’78]. This algorithm is not applicable in the streaming setting.

- **Threshold-Greedy**: The output is the $k$ data points provided by Threshold-Greedy [BV’14].
  - Maintains a continuously decreasing threshold and takes elements when their marginal value is above the threshold.

- **Sample-Greedy**: The output is the $k$ data points produced by applying Lazy-Greedy on a subset of data points parametrized by sampling probability $p$.

- **Random selection**: The output is $k$ randomly selected data points from $V$. 
Nonparametric Regression

Parkinson dataset consists of 5,875 data points with 22 attributes.

Similar utility but orders of magnitude faster!
Exemplar Based Clustering

A set of 10,000 Images with 3072 attributes.

Similar utility but orders of magnitude faster!
When the utility functions $f$ is (non-monotone) submodular,

$$S^* = \arg \max_{S \in \mathcal{I}} f(S)$$

and there are constraints imposed by the data summarization application.
Double Greedy: Unconstrained

- **Unconstrained non-monotone** submodular maximization:
  \[ S^* = \arg \max_{S \subseteq \Omega} f(S) \]

- There is no restriction on the choice of \( S \)
Random Double Greedy

The random double greedy algorithm, in expectation, provides a 1/2 approximation guarantee for the unconstrained non-monotone submodular maximization problem:

$$E[f(S_{\text{random double greedy}})] \geq (1/2) f(S^*)$$

“A Tight Linear Time (1/2)-Approximation for Unconstrained Submodular Maximization”, FOCS’12
(Buchbinder, Feldman, Naor, Schwartz)
Random Greedy: Constrained

- **Constrained non-monotone** submodular maximization:

\[
S^* = \arg\max_{S \subseteq \Omega, |S| \leq k} f(S)
\]

- Choose a set \( S \) of size at most \( k \)
Sample Greedy: Constrained

- **Constrained non-monotone** submodular maximization:
  \[ S^* = \arg \max_{S \in \mathcal{I}} f(S) \]

- More general constraints such as a \( \rho \)-extendible system \( \mathcal{I} \)

(Feldman, Harshaw, Karbasi)
Algorithm FANTOM

Input: Set $E$, a membership oracle for $p$-system $I \subseteq 2^E$, and $l$ knapsack-cost functions $c_i : E \rightarrow [0, 1]$.

Output: Set $S$ satisfying $S \subseteq I$ and $c_i(S) \leq 1 \ \forall i$.

1: $S = \emptyset$.
2: $M = \max_{j \in E} f(j)$, $\gamma = 2pM/(p+1)(2p+1)$, $U = \emptyset$.
3: $R = \{\gamma, (1+\varepsilon)\gamma, (1+\varepsilon)^2\gamma, (1+\varepsilon)^3\gamma, \ldots, \gamma^n\}$.
4: For $\rho \in R$ do
5: \[ \Omega = E. \]
6: For $i = 1; i \leq p+1; i ++$ do
7: \[ S_i = \text{run greedy and at each step pick} \]
\[ \text{the element if and only if } f_S(j)/\sum_{i=1}^l c_{ij} \geq \rho \]
8: \[ S_i = \text{argmax} (S_i, \text{argmax}_{z \in E} f(z)) \]
9: \[ S'_{i} = \text{Unconstrained-Maximization}(S_{i}) \]
10: \[ \Omega = \Omega - S_i \]
11: \[ S = \text{argmax}(S, S_i) \]
12: return $S$

First efficient algorithm for non-monotone submodular max s.t a $p$-system and $l$-knapsack

Approximation ratio

$f(S) \geq (1 + \varepsilon)(p + 1)(2p + 2l + 1)/p \ \text{OPT}$

Running time

$O(nrp \log(n)/\varepsilon)$

“Fast Constrained Submodular Maximization”, ICML’16
(Mirzasoleiman, Badanidiyuru, Karbasi)
Given a monotone submodular function $F$, and a cardinality constraint $K$, find a set $A^*$ such that

$$A^* = \arg \max_{|A| \leq k} F(A)$$

Centralized Solution

Streaming Solution

Distributed Solution
Streaming Solution

Data *cannot* be loaded on the main memory
Data Stream
Problem: Greedy can be impractical

- Greedy approaches require random access to the complete data set; but for truly large-scale problems, where data is residing on disk, or arriving over time at a fast pace they are often impractical.

Is it possible to summarize a massive data set “on the fly” i.e., at any point of time we have access only to a small fraction of data?
Stream-Greedy

Can we swap $e$ with any elements in $S$ s.t. the utility increases?

$$\exists x \in S \text{ s.t. } f(S \cup \{e\} \setminus \{x\}) > f(S)$$
Stream-Greedy is Bad!

- **Stream-Greedy** could be arbitrary poor!

- Example: a set $X$ and a collection $V$ of subsets of $X$:

  $$S^* = \{\{1,2,\ldots,k\}, \{k+1\}, \{k+2\}, \ldots, \{2k\}, \ldots, \{k^2+1\}, \ldots, \{k^2+k\}\}$$

$$f(S) = \sum_{x \in \bigcup_{v \in S} v} w(x)$$

$$w(1) = \cdots = w(k) = 1$$
$$w(k+1) = \cdots = w(2k) = 1 + \epsilon$$
$$\ldots$$
$$w(k^2+1) = \cdots = w(k^2+k) = 1 + k\epsilon$$

and $\epsilon \ll 1$

$$f(S^*) \approx k^2 \text{ and } f(S) \approx k$$
Streaming with Preemption

For monotonic submodular functions with a cardinality constraint $k$, the streaming algorithm with preemption gives a solution $S_{\text{preemption}}$ such that:

$$f(S_{\text{preemption}}) \geq (1/4) f(S^*)$$

“Online Submodular Maximization with Preemption.”, SODA’15

Choose the one with largest benefit to exchange

(Buchbinder, Feldman, Schwartz)
Data Summarization on the Fly!

Data Stream

\[ OPT \geq v \geq \alpha \text{OPT} \]

\[ \Delta_f(e_i|S) \geq (v/2 - f(S))/(k - |S|) \]

\[ f\left(\left\{ \right\}\right) \rightarrow \text{max} \]

\( \alpha \text{OPT}/2 \)

Properties:
- 1 pass
- \( f(S) \geq \alpha \text{OPT}/2 \)
- \( O(k) \) memory
- \( O(1) \) update time
Knowing max marginal is enough

- But how can we find a good approximation of $\text{OPT}$?
Obtaining $m$ requires a full pass over the data 😞
Algorithm SIEVE-STREAMING

1: $O = \{(1 + \epsilon)^i | i \in \mathbb{Z}\}$
2: For each $v \in O$, $S_v := \emptyset$ (maintain the sets only for the necessary $v$’s lazily)
3: $m = 0$
4: for $i = 1$ to $n$ do
5:     $m := \max(m, f(\{e_i\}))$
6: $O_i = \{(1 + \epsilon)^i | m \leq (1 + \epsilon)^i \leq 2 \cdot k \cdot m\}$
7: Delete all $S_v$ such that $v \notin O_i$.
8: for $v \in O_i$ do
9:     if $\Delta f(e_i | S_v) \geq \frac{v}{2} - \frac{f(S_v)}{k - |S_v|}$ and $|S_v| < k$ then
10:        $S_v := S_v \cup \{e_i\}$
11: return $\arg\max_{v \in O_n} f(S_v)$

First streaming algorithm for cardinality-constrained submodular maximization with
- Constant factor approximation guarantee $f(S) \geq (1/2 - \epsilon) \cdot \text{OPT}$
- Makes no assumptions on the data stream
- Requires only a single pass
- Only $O(k \log k)$ memory
- Only $O(\log k)$ update time
- Assuming nothing but monotone submodularity

“Streaming Submodular Maximization: Massive Data Summarization on the Fly”, KDD’14
(Badanidiyuru, Mirzasoleiman, Karbasi, Krause)
Experiments

- **Benchmarks:**
  
  - **Stream-Greedy**: The output is the $k$ data points provided by Stream-Greedy [GK’10].
    - For each new data point, we check whether switching it with an element in $S$ will increase the value of the utility function $f$.
  
  - **Random selection**: the output is $k$ randomly selected data points from $V$. 
Census data set consists of 2,458,285 data points with 68 attributes.

Similar utility but orders of magnitude faster!
Nonparametric Regression

*Yahoo! Webscope* data set consists of 45,811,883 user visits from the Featured Tab of the Today Module on the Yahoo! Front Page.

Similar utility but orders of magnitude faster!
**Constrained Non-Monotone Submodular Maximization**

"Do Less, Get More: Streaming Submodular Maximization with Subsampling", arXiv’18

(Feldman, Karbasi, Kazemi)
Given a monotone submodular function $F$, and a cardinality constraint $K$, find a set $A^*$ such that

$$A^* = \arg \max_{|A| \leq k} F(A)$$

Centralized Solution

Streaming Solution

Distributed Solution
Distributed Solution

Big data but also many servers
Problem: Scale Up

- On massive data (80,000,000 images) the greedy policies take a few days/weeks to complete.

- Can we parallelise the greedy approach?
Two-Stage Greedy (GreeDi)

DATA

Greedy(k) → Greedy(k) → Greedy(k) → Greedy(k) → Greedy(k)
Theoretical Guarantees

• Split data (arbitrarily) among \( m \) machine

GreeDi

For monotonic submodular functions, GreeDi gives

\[
F(A) \geq \frac{1 - 1/e}{\sqrt{\min(m (m/k) k) \cdot \Theta PT}} \cdot \Theta OPT
\]

Cannot do better in **general**

“Distributed Submodular Cover: Succinctly Summarizing Massive Data”, NIPS’15

(Mirzasoleiman, Karbasi, Badanidiyuru, Krause)

“Distributed Submodular Maximization: Identifying Representative Elements in Massive Data”, NIPS’13

(Mirzasoleiman, Karbasi, Sarkar, Krause)
Two-Stage Greedy (Random Splitting)
GreeDi + Random Partition

Randomization helps! 😊

**GreeDi**

For monotonic submodular functions, GreeDi gives and random partitioning of data on m machines

\[ \mathbb{E}[F(A)] \geq \frac{1 - 1/e}{2} F(A^*) \]

Can do better with **randomization**

“The Power of Randomization: Distributed Submodular Maximization on Massive datasets”, ICML’15

Barbosa, Ene, Nguyen, Ward

“Randomized Composable Core-sets for Distributed Submodular Maximization”, STOC’15

Mirrokni, Zadimoghaddam
Nonparametric Regression on Hadoop

- 45,811,833 user visits on Yahoo! front page
Exemplar-Based Clustering on 80M Images (Hadoop)
Exemplar-Based Clustering on 80M Images (Hadoop)
Exemplar-Based Clustering on 80M Images (Hadoop)
Exemplar-Based Clustering on 80M Images (Hadoop)
Conclusion

- Big data is getting much BIGGER
- Summarization is inevitable
- Submodularity provides a unifying framework
- We have now fast centralised, streaming, and distributed methods to (approximately) optimise submodular functions
Thank You