Submodularity in Machine Learning

Stefanie Jegelka
BigData@CSAIL Meeting
Nov 6, 2015
Set functions

ground set

\[ V = \{ \text{food items} \} \]

\[ F : 2^V \rightarrow \mathbb{R} \]

\[ F \left( \begin{array}{c} \text{fries} \\ \text{so} \end{array} \right) = \text{cost of buying items together, or} \]

\[ \text{utility, or} \]

\[ \text{probability, ...} \]
Same machinery for all of those?

Formalization: min / max set function generally: very hard 😞

Exponential yet tractable?

special properties help!  Submodularity
Sensing / Information Gathering

\[ \mathcal{V} = \text{all possible locations} \]
\[ F(S) = \text{information gained from locations in } S \]
Marginal gain

• Given set function \( F : 2^V \rightarrow \mathbb{R} \)

• Marginal gain: \( F(s|A) = F(A \cup \{s\}) - F(A) \)
Diminishing gains

placement A = \{1,2\}

placement B = \{1,2,3,4\}

Big gain

for all \(A \subseteq B\)

and s not in B

\[ F(A \cup s) - F(A) \geq F(B \cup s) - F(B) \]
Examples: utility

maximize information / coverage

maximize coverage & diversity

maximize influence

find the common

- (Krause & Guestrin 2005)
Submodularity

\[ A \subseteq B \]

\[ F(A \cup s) - F(A) \geq F(B \cup s) - F(B) \]

extra cost: one drink

diminishing marginal costs

extra cost: free refill 😊
Examples: cost / energy

\[ F(S) = \sum_{e \in \text{cut}} w(e) \]

\[ \text{loss}(w) + F(S) \]

\[ \text{S} = \text{support}(w) \]

\( S \subseteq V \)

Minimize incoherence/
maximize coherence

MAP inference

\[ \max_x P(x \mid I) \]

spatial/temporal coherence

(\text{Ising, Potts, Porteous et al., ...})

\[ P(x = 1_s) \propto \exp(-F(S)) \]
Success stories

- Water sensor networks
- Summarization
- Segmentation
- ...

The Battle of the Water Sensor Networks (BWSN): A Design Challenge for Engineers and Algorithms

Avi Ostfeld1; James G. Uber2; Elad Salomons3; Jonathan Jean-Paul Watson4; Gianluca Dorin5; Philip Jonkergou Soon-Thiam Khu6; Dragan Savic7; Demetrios Ellades8; Brian D. Barkdoll9; Roberto Gueli10; Jinhui J. Huang11; Andreas Kraute12; Jure Leskovec13; Shannon Isovi14; Jeanne VanBriesen15; Mitchell Small16; Paul Fischbeck17; Gary B. Trachtman18; Zheng Yi19

Abstract: Following the events of September 11, 2001, in the United States, water supply systems have increased dramatically. Among the different threats is a deliberate chemical or biological contaminant injection, due to both...
Why submodularity useful?

- very general condition, easy to check: many many applications!

- fast algorithms with guarantees:
  1. Minimize $F(S)$: find a set with minimum cost
     convex optimization
Why submodularity useful?

- very general condition, often easy to check: many many applications!

- fast algorithms with guarantees:
  1. Minimize $F(S)$: find a set with minimum cost
     convex optimization
  2. Maximize $F(S)$: find a set with maximum utility
     greedy methods
Submodularity at scale?

convergence speed (img labeling, denoising, ...)

iteration (time)

error

iteration (time)

Speedup for Max Graph Cut

Ideal

CC2G, IT−2004

CF2G, IT−2004

CC2G, ZigZag

CF2G, ZigZag

# threads

41M Vertices, 1.1B Edges

ZigZag: 25M Vertices, 28 Edges
Scalable maximization

- Parallel double greedy maximization
  
pick in parallel, communicate only when necessary
  preserves serial theoretical guarantees
  “always correct, almost always fast”
  (Pan, Jegelka, Gonzalez, Bradley, Jordan 2014)

- other schemes: distributed, stochastic, filtering

Pick the best of m+1 solutions
Submodularity in Machine Learning

- very general, intuitive condition
  - many many applications!
- fast algorithms with guarantees!
  1. Minimization: minimum cost
     - convex optimization
  2. Maximization: maximum utility
     - greedy algorithms
- scalability: recent progress