Submodular Functions, Optimization, and Applications to Machine Learning

— Spring Quarter, Lecture 1 —

http://j.ee.washington.edu/~bilmes/classes/ee596b_spring_2014/

Prof. Jeff Bilmes

University of Washington, Seattle
Department of Electrical Engineering

http://melodi.ee.washington.edu/~bilmes

Mar 31st, 2014

\[
f(A) + f(B) \geq f(A \cup B) + f(A \cap B)
\]

Clockwise from top left:
- László Lovász
- Jack Edmonds
- Satoru Fujishige
- George Nemhauser
- Laurence Wolsey
- András Frank
- Lloyd Shapley
- H. Narayanan
- Robert Bixby
- William Cunningham
- William Tutte
- Richard Rado
- Alexander Schrijver
- Garrett Birkhoff
- Hassler Whitney
- Richard Dedekind
Announcements

- Welcome to the class!
- Submodular Functions, Optimization, and Applications to Machine Learning, EE596B.
- Mueller 154
- Weekly Office Hours: Wednesdays, 3:30-4:30, 10 minutes after class ends on Wednesdays.
- Class web page is at our web page (http://j.ee.washington.edu/~bilmes/classes/ee596b_spring_2014/)
This course will serve as an introduction to submodular functions including methods for their optimization, and how they have been (and can be) applied in many application domains.
Rough Outline

- Introduction to submodular functions, including definitions, real-world and contrived examples of submodular functions, properties, operations that preserve submodularity, submodular variants and special submodular functions, and computational properties.

- Background on submodular functions, including a brief overview of the theory of matroids and lattices.

- Polyhedral properties of submodular functions

- The Lovász extension of submodular functions. The Choquet integral.

- Submodular maximization algorithms under simple constraints, submodular cover problems, greedy algorithms, approximation guarantees
Rough Outline (cont. II)

- Submodular minimization algorithms, a history of submodular minimization, including both numerical and combinatorial algorithms, computational properties of these algorithms, and descriptions of both known results and currently open problems in this area.

- Submodular flow problems, the principle partition of a submodular function and its variants.

- Constrained optimization problems with submodular functions, including maximization and minimization problems with various constraints. An overview of recent problems addressed in the community.

- Applications of submodularity in computer vision, constraint satisfaction, game theory, information theory, norms, natural language processing, graphical models, and machine learning
Classic References

Useful Books

- Fujishige, “Submodular Functions and Optimization”, 2005
- Narayanan, “Submodular Functions and Electrical Networks”, 1997
- Schrijver, “Combinatorial Optimization”, 2003
- Additional readings that will be announced here.
Recent online material (some with an ML slant)

- Previous version of this class http://j.ee.washington.edu/~bilmes/classes/ee596a_fall_2012/.
- Francis Bach's updated 2013 text. http://hal.archives-ouvertes.fr/docs/00/87/06/09/PDF/submodular_fot_revised_hal.pdf
- Tom McCormick's overview paper on submodular minimization http://people.commerce.ubc.ca/faculty/mccormick/sfmchap8a.pdf
- Georgia Tech's 2012 workshop on submodularity: http://www.arc.gatech.edu/events/arc-submodularity-workshop
Facts about the class

- **Prerequisites**: ideally knowledge in probability, statistics, convex optimization, and combinatorial optimization these will be reviewed as necessary. The course is open to students in all UW departments. Any questions, please contact me.

- **Text**: We will be drawing from the book by Satoru Fujishige entitled "Submodular Functions and Optimization" 2nd Edition, 2005, but we will also be reading research papers that will be posted here on this web page, especially for some of the application areas.

- **Grades and Assignments**: Grades will be based on a combination of a final project (45%), homeworks (55%). There will be between 3-6 homeworks during the quarter.

- **Final project**: The final project will consist of a 4-page paper (conference style) and a final project presentation. The project must involve using/dealing mathematically with submodularity in some way or another.
Facts about the class

- Homework must be submitted electronically using our assignment dropbox (https://canvas.uw.edu/courses/895956/assignments). PDF submissions only please. Photos of neatly hand written solutions, combined into one PDF, are fine.

- Lecture slides - are being prepared as we speak. I will try to have them up on the web page the night before each class. I will not only draw from the book but other sources which will be listed at the end of each set of slides.

- Slides from previous version of this class are at http://j.ee.washington.edu/~bilmes/classes/ee596a_fall_2012/.
Other logistics

- Almost all equations will have numbers.
Almost all equations will have numbers.

Equations will be numbered with lecture number, and number within lecture in the form \((\ell.j)\) where \(\ell\) is the lecture number and \(j\) is the \(j\)th equation in lecture \(\ell\). For example,

\[
f(A) = f(V \setminus A) \tag{1.1}
\]

By the way \(V \setminus A \equiv \{v \in V : v \notin A\}\) is set subtraction, sometimes written as \(V - A\).
Other logistics

- Almost all equations will have numbers.
- Equations will be numbered with lecture number, and number within lecture in the form \((\ell.j)\) where \(\ell\) is the lecture number and \(j\) is the \(j^{th}\) equation in lecture \(\ell\). For example,
  \[
  f(A) = f(V \setminus A) \tag{1.1}
  \]
  By the way \(V \setminus A \equiv \{v \in V : v \notin A\}\) is set subtraction, sometimes written as \(V - A\).
- Theorems, Lemmas, postulates, etc. will be numbered with \((\ell.s.j)\) where \(\ell\) is the lecture number, \(s\) is the section number, and \(j\) is the order within that section.
Other logistics

- Almost all equations will have numbers.
- Equations will be numbered with lecture number, and number within lecture in the form \((\ell.j)\) where \(\ell\) is the lecture number and \(j\) is the \(j^{th}\) equation in lecture \(\ell\). For example,
  \[
  f(A) = f(V \setminus A) \tag{1.1}
  \]
  By the way \(V \setminus A \equiv \{v \in V : v \not\in A\}\) is set subtraction, sometimes written as \(V - A\).
- Theorems, Lemmas, postulates, etc. will be numbered with \((\ell.s.j)\) where \(\ell\) is the lecture number, \(s\) is the section number, and \(j\) is the order within that section.

### Theorem 1.1.1 (foo’s theorem)

foo
Other logistics

- Almost all equations will have numbers.
- Equations will be numbered with lecture number, and number within lecture in the form \((\ell.j)\) where \(\ell\) is the lecture number and \(j\) is the \(j^{th}\) equation in lecture \(\ell\). For example,

\[
f(A) = f(V \setminus A) \tag{1.1}
\]

By the way \(V \setminus A \equiv \{v \in V : v \notin A\}\) is set subtraction, sometimes written as \(V - A\).

- Theorems, Lemmas, postulates, etc. will be numbered with \((\ell.s.j)\) where \(\ell\) is the lecture number, \(s\) is the section number, and \(j\) is the order within that section.

**Theorem 1.1.1 (foo’s theorem)**

foo

- Exception to these rules is in the review sections, where theorems, equation, etc. (even if repeated) will have new reference numbers.
Read chapter 1 from Fujishige book.
Announcements, Assignments, and Reminders

Please do use our discussion board (https://canvas.uw.edu/courses/895956/discussion_topics) for all questions, comments, so that all will benefit from them being answered.
L1 (3/31): Motivation, Applications, & Basic Definitions
L2: (4/2): Applications, Basic Definitions, Properties
L3: 
L4: 
L5: 
L6: 
L7: 
L8: 
L9: 
L10: 

L11: 
L12: 
L13: 
L14: 
L15: 
L16: 
L17: 
L18: 
L19: 
L20: 

Finals Week: June 9th-13th, 2014.
This is where each day we will be reviewing previous lecture material.
Submodular Definitions

Definition 1.3.1 (submodular concave)

A function $f : 2^V \rightarrow \mathbb{R}$ is submodular if for any $A, B \subseteq V$, we have that:

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B) \tag{1.2}$$

An alternate and (as we will soon see) equivalent definition is:

Definition 1.3.2 (diminishing returns)

A function $f : 2^V \rightarrow \mathbb{R}$ is submodular if for any $A \subseteq B \subset V$, and $v \in V \setminus B$, we have that:

$$f(A \cup \{v\}) - f(A) \geq f(B \cup \{v\}) - f(B) \tag{1.3}$$

This means that the incremental “value”, “gain”, or “cost” of $v$ decreases (diminishes) as the context in which $v$ is considered grows from $A$ to $B$. 
Sets and set functions

We are given a finite “ground” set of objects:

\[ V = \{ \text{various objects} \} \]

Also given a set function \( f : 2^V \rightarrow \mathbb{R} \) that valuates subsets \( A \subseteq V \).
Ex: \( f(V) = 6 \)
Sets and set functions

Subset $A \subseteq V$ of objects:

Also given a set function $f : 2^V \rightarrow \mathbb{R}$ that valuates subsets $A \subseteq V$.
Ex: $f(A) = 1$
Sets and set functions

Subset $B \subseteq V$ of objects:

Also given a set function $f : 2^V \to \mathbb{R}$ that valuates subsets $A \subseteq V$.
Ex: $f(B) = 6$
We are given a finite set of objects $V$ of size $n = |V|$. 

Discrete Optimization Problems

- We are given a finite set of objects $V$ of size $n = |V|$. 

Alternative, we may minimize rather than maximize.
Discrete Optimization Problems

- We are given a finite set of objects $V$ of size $n = |V|$.
- We consider subsets of $V$. There are $2^n$ such subsets (denoted $2^V$).
Discrete Optimization Problems

- We are given a finite set of objects $V$ of size $n = |V|$.
- We consider subsets of $V$. There are $2^n$ such subsets (denoted $2^V$).
- We have a function $f : 2^V \to \mathbb{R}$ that judges the quality (or value, or cost, or etc.) of each subset.
Discrete Optimization Problems

- We are given a finite set of objects $V$ of size $n = |V|$.
- We consider subsets of $V$. There are $2^n$ such subsets (denoted $2^V$).
- We have a function $f : 2^V \rightarrow \mathbb{R}$ that judges the quality (or value, or cost, or etc.) of each subset.
- We may be interested only in a subset of the set of possible subsets, namely $S \subseteq 2^V$. E.g., $S = \{ S \subseteq V : |S| \leq k \}$. The set of sets $S$ might or might not itself be a function of $f$ (e.g., $S = \{ S \subseteq V : f(S) \leq \alpha \}$).
Discrete Optimization Problems

- We are given a finite set of objects \( V \) of size \( n = |V| \).
- We consider subsets of \( V \). There are \( 2^n \) such subsets (denoted \( 2^V \)).
- We have a function \( f : 2^V \rightarrow \mathbb{R} \) that judges the quality (or value, or cost, or etc.) of each subset.
- We may be interested only in a subset of the set of possible subsets, namely \( S \subseteq 2^V \). E.g., \( S = \{ S \subseteq V : |S| \leq k \} \). The set of sets \( S \) might or might not itself be a function of \( f \) (e.g., \( S = \{ S \subseteq V : f(S) \leq \alpha \} \)).
- A general discrete optimization problem we consider here is:

\[
\begin{align*}
\text{maximize} & \quad f(S) \\
\text{subject to} & \quad S \in S
\end{align*}
\]
Discrete Optimization Problems

- We are given a finite set of objects $V$ of size $n = |V|$.
- We consider subsets of $V$. There are $2^n$ such subsets (denoted $2^V$).
- We have a function $f : 2^V \rightarrow \mathbb{R}$ that judges the quality (or value, or cost, or etc.) of each subset.
- We may be interested only in a subset of the set of possible subsets, namely $S \subseteq 2^V$. E.g., $S = \{S \subseteq V : |S| \leq k\}$. The set of sets $S$ might or might not itself be a function of $f$ (e.g., $S = \{S \subseteq V : f(S) \leq \alpha\}$).
- A general discrete optimization problem we consider here is:

$$\text{maximize } S \subseteq 2^V f(S)$$
subject to $S \in S$ \hspace{1cm} (1.4)

- Alternatively, we may minimize rather than maximize.
Set functions are pseudo-Boolean functions

- Any set $A \subseteq V$ can be represented as a binary vector $x \in \{0, 1\}^V$ (a “bit vector” representation of a set).
Set functions are pseudo-Boolean functions

- Any set $A \subseteq V$ can be represented as a binary vector $x \in \{0, 1\}^V$ (a "bit vector" representation of a set).
- The characteristic vector of a set is given by $1_A \in \{0, 1\}^V$ where for all $v \in V$, we have:

\[
1_A(v) = \begin{cases} 
1 & \text{if } v \in A \\
0 & \text{else}
\end{cases}
\]  

(1.5)
Set functions are pseudo-Boolean functions

- Any set \( A \subseteq V \) can be represented as a binary vector \( x \in \{0, 1\}^V \) (a “bit vector” representation of a set).
- The characteristic vector of a set is given by \( 1_A \in \{0, 1\}^V \) where for all \( v \in V \), we have:

\[
1_A(v) = \begin{cases} 
1 & \text{if } v \in A \\
0 & \text{else} 
\end{cases}
\]  

(1.5)

- It is sometimes useful to go back and forth between \( X \) and \( x(X) \triangleq 1_X \).
Set functions are pseudo-Boolean functions

- Any set $A \subseteq V$ can be represented as a binary vector $x \in \{0, 1\}^V$ (a “bit vector” representation of a set).
- The characteristic vector of a set is given by $1_A \in \{0, 1\}^V$ where for all $v \in V$, we have:

$$1_A(v) = \begin{cases} 1 & \text{if } v \in A \\ 0 & \text{else} \end{cases}$$

- It is sometimes useful to go back and forth between $X$ and $x(X) \triangleq 1_X$.
- $f(x) : \{0, 1\}^V \rightarrow \mathbb{R}$ is a pseudo-Boolean function, and submodular functions are a special case.
Ignoring how complex and general this problem can be for the moment, let's consider some possible applications.

In the rest of this section of slides, we will see many seemingly different applications that, ultimately, you will all hopefully see are strongly related to submodularity.

We'll see, submodularity is as common and natural for discrete problems as is convexity for continuous problems.
Example Discrete Optimization Problems

- **Combinatorial Problems**: e.g., set cover, max $k$ coverage, vertex cover, edge cover, graph cut problems.
- **Operations Research**: facility location (uncapacitated)
- **Sensor placement**
- **Information**: Information gain and feature selection, information theory
- **Mathematics**: e.g., monge matrices
- **Networks**: Social networks, influence, viral marketing, information cascades, diffusion networks
- **Graphical models**: tree distributions, factors, and image segmentation
- **Diversity** and its models
- **NLP**: Natural language processing: document summarization, web search, information retrieval
- **ML**: Machine Learning: active/semi-supervised learning
- **Economics**: markets, economies of scale
We are given a finite set \( V \) of \( n \) elements and a set of subsets \( \mathcal{V} = \{V_1, V_2, \ldots, V_m\} \) of \( m \) subsets of \( V \), so that \( V_i \subseteq V \) and \( \bigcup_i V_i = V \).
We are given a finite set $V$ of $n$ elements and a set of subsets $\mathcal{V} = \{V_1, V_2, \ldots, V_m\}$ of $m$ subsets of $V$, so that $V_i \subseteq V$ and $\bigcup_i V_i = V$.

The goal of minimum SET COVER is to choose the smallest subset $A \subseteq [m] \triangleq \{1, \ldots, m\}$ such that $\bigcup_{a \in A} V_a = V$. 
Set Cover and Maximum Coverage

- We are given a finite set $V$ of $n$ elements and a set of subsets $\mathcal{V} = \{V_1, V_2, \ldots, V_m\}$ of $m$ subsets of $V$, so that $V_i \subseteq V$ and $\bigcup_i V_i = V$.

- The goal of minimum Set Cover is to choose the smallest subset $A \subseteq [m] \triangleq \{1, \ldots, m\}$ such that $\bigcup_{a \in A} V_a = V$.

- Maximum $k$ cover: The goal in Maximum Coverage is, given an integer $k \leq m$, select $k$ subsets, say $\{a_1, a_2, \ldots, a_k\}$ with $a_i \in [m]$ such that $|\bigcup_{i=1}^k V_{a_i}|$ is maximized.
Set Cover and Maximum Coverage

- We are given a finite set $V$ of $n$ elements and a set of subsets $\mathcal{V} = \{V_1, V_2, \ldots, V_m\}$ of $m$ subsets of $V$, so that $V_i \subseteq V$ and $\bigcup_i V_i = V$.

- The goal of minimum Set Cover is to choose the smallest subset $A \subseteq [m] \triangleq \{1, \ldots, m\}$ such that $\bigcup_{a \in A} V_a = V$.

- Maximum $k$ cover: The goal in Maximum Coverage is, given an integer $k \leq m$, select $k$ subsets, say $\{a_1, a_2, \ldots, a_k\}$ with $a_i \in [m]$ such that $|\bigcup_{i=1}^k V_{a_i}|$ is maximized.

- Both Set cover and maximum coverage are well known to be NP-hard, but have a fast greedy approximation algorithm.
Definition 1.5.1 (vertex cover)

A vertex cover (an “vertex-based cover of edges”) in graph $G = (V, E)$ is a set $S \subseteq V(G)$ of vertices such that every edge in $G$ is incident to at least one vertex in $S$.

- Let $I(S)$ be the number of edges incident to vertex set $S$. Then we wish to find the smallest set $S \subseteq V$ subject to $I(S) = |E|$.

Definition 1.5.2 (edge cover)

A edge cover (an “edge-based cover of vertices”) in graph $G = (V, E)$ is a set $F \subseteq E(G)$ of edges such that every vertex in $G$ is incident to at least one edge in $F$.

- Let $|V|(F)$ be the number of vertices incident to edge set $F$. Then we wish to find the smallest set $F \subseteq E$ subject to $|V|(F) = |V|$.
**Minimum cut:** Given a graph $G = (V, E)$, find a set of vertices $S \subseteq V$ that minimize the cut (set of edges) between $S$ and $V \setminus S$. 
Graph Cut Problems

- **Minimum cut**: Given a graph $G = (V, E)$, find a set of vertices $S \subseteq V$ that minimize the cut (set of edges) between $S$ and $V \setminus S$.

- **Maximum cut**: Given a graph $G = (V, E)$, find a set of vertices $S \subseteq V$ that minimize the cut (set of edges) between $S$ and $V \setminus S$. 
Graph Cut Problems

- **Minimum cut**: Given a graph $G = (V, E)$, find a set of vertices $S \subseteq V$ that minimize the cut (set of edges) between $S$ and $V \setminus S$.

- **Maximum cut**: Given a graph $G = (V, E)$, find a set of vertices $S \subseteq V$ that minimize the cut (set of edges) between $S$ and $V \setminus S$.

- Let $f : 2^V \to \mathbb{R}_+$ be the cut function, namely for any given set of nodes $X \subseteq V$, $f(X)$ measures the number of edges between nodes $X$ and $V \setminus X$. 
Graph Cut Problems

- **Minimum cut**: Given a graph $G = (V, E)$, find a set of vertices $S \subseteq V$ that minimize the cut (set of edges) between $S$ and $V \setminus S$.

- **Maximum cut**: Given a graph $G = (V, E)$, find a set of vertices $S \subseteq V$ that minimize the cut (set of edges) between $S$ and $V \setminus S$.

- Let $f : 2^V \to \mathbb{R}_+$ be the cut function, namely for any given set of nodes $X \subseteq V$, $f(X)$ measures the number of edges between nodes $X$ and $V \setminus X$.

- Weighted versions, where rather than count, we sum the (non-negative) weights of the edges of a cut.
Graph Cut Problems

- **Minimum cut:** Given a graph $G = (V, E)$, find a set of vertices $S \subseteq V$ that minimize the cut (set of edges) between $S$ and $V \setminus S$.
- **Maximum cut:** Given a graph $G = (V, E)$, find a set of vertices $S \subseteq V$ that minimize the cut (set of edges) between $S$ and $V \setminus S$.
- Let $f : 2^V \rightarrow \mathbb{R}_+$ be the cut function, namely for any given set of nodes $X \subseteq V$, $f(X)$ measures the number of edges between nodes $X$ and $V \setminus X$.
- Weighted versions, where rather than count, we sum the (non-negative) weights of the edges of a cut.
- Many examples of this, we will see more later.
Facility/Plant Location (uncapacitated)

- Core problem in operations research and strong early motivation for submodular functions.
- Goal: as efficiently as possible, place “facilities” (factories) at certain locations to satisfy sites (at all locations) having various demands.
Facility/Plant Location (uncapacitated)

- Core problem in operations research and strong early motivation for submodular functions.
- Goal: as efficiently as possible, place “facilities” (factories) at certain locations to satisfy sites (at all locations) having various demands.
Facility/Plant Location (uncapacitated)

- Core problem in operations research and strong early motivation for submodular functions.
- Goal: as efficiently as possible, place “facilities” (factories) at certain locations to satisfy sites (at all locations) having various demands.

```
facility locations       sites
1
2
3 m_3
4
5
```

```
1
2
3
4
```

\[ C_{2A} \]
Facility/Plant Location (uncapacitated)

- Let $F = \{1, \ldots, f\}$ be a set of possible factory/plant locations for facilities to be built.
- $S = \{1, \ldots, s\}$ is a set of sites needing to be serviced (e.g., cities, clients).
- Let $c_{ij}$ be the “benefit” (e.g., $1/c_{ij}$ is the cost) of servicing site $i$ with facility location $j$.
- Let $m_j$ be the benefit (e.g., either $1/m_j$ is the cost or $-m_j$ is the cost) to build a plant at location $j$.
- Each site needs to be serviced by only one plant but no less than one.
- Define $f(A)$ as the “delivery benefit” plus “construction benefit” when the locations $A \subseteq F$ are to be constructed.
- We can define $f(A) = \sum_{j \in A} m_j + \sum_{i \in F} \max_{j \in A} c_{ij}$.
- Goal is to find a set $A$ that maximizes $f(A)$ (the benefit) placing a bound on the number of plants $A$ (e.g., $|A| \leq k$).
Sensor Placement

- Given an environment, there is a set $V$ of candidate locations for placement of a sensor (e.g., temperature, gas, audio, video, bacteria or other environmental contaminant, etc.).
Sensor Placement

- Given an environment, there is a set $V$ of candidate locations for placement of a sensor (e.g., temperature, gas, audio, video, bacteria or other environmental contaminant, etc.).

- We have a function $f(S)$ that measures the “coverage” of any given set $S$ of sensor placement decisions. Then $f(V)$ is maximum possible coverage.
Sensor Placement

- Given an environment, there is a set $V$ of candidate locations for placement of a sensor (e.g., temperature, gas, audio, video, bacteria or other environmental contaminant, etc.).
- We have a function $f(S)$ that measures the “coverage” of any given set $S$ of sensor placement decisions. Then $f(V)$ is maximum possible coverage.
- One possible goal: choose smallest set $S$ such that $f(S) = \alpha f(V)$ with $0 < \alpha \leq 1$. 

Sensor Placement

- Given an environment, there is a set $V$ of candidate locations for placement of a sensor (e.g., temperature, gas, audio, video, bacteria or other environmental contaminant, etc.).
- We have a function $f(S)$ that measures the “coverage” of any given set $S$ of sensor placement decisions. Then $f(V)$ is maximum possible coverage.
- One possible goal: choose smallest set $S$ such that $f(S) = \alpha f(V)$ with $0 < \alpha \leq 1$.
- Another possible goal: choose size at most $k$ set $S$ such that $f(S)$ is maximized.
Sensor Placement

- Given an environment, there is a set $V$ of candidate locations for placement of a sensor (e.g., temperature, gas, audio, video, bacteria or other environmental contaminant, etc.).

- We have a function $f(S)$ that measures the “coverage” of any given set $S$ of sensor placement decisions. Then $f(V)$ is maximum possible coverage.

- One possible goal: choose smallest set $S$ such that $f(S) = \alpha f(V)$ with $0 < \alpha \leq 1$.

- Another possible goal: choose size at most $k$ set $S$ such that $f(S)$ is maximized.

- Environment could be a floor of a building, water network, monitored ecological preservation.
Sensor Placement within Buildings

- An example of a room layout. Should be possible to determine temperature at all points in the room. Sensors cannot sense beyond wall (thick black line) boundaries.
Sensor Placement within Buildings

- Example sensor placement using small range cheap sensors (located at red dots).
Sensor Placement within Buildings

- Example sensor placement using longer range expensive sensors (located at red dots).
Sensor Placement within Buildings

- Example sensor placement using mixed range sensors (located at red dots).
Information Gain and Feature Selection

- **Task**: pattern recognition based on (at most) features $X_V$ to predict random variable $Y$. True model is $p(Y, X_V)$, where $V$ is a finite set of feature indices.
Information Gain and Feature Selection

- Task: pattern recognition based on (at most) features $X_V$ to predict random variable $Y$. True model is $p(Y, X_V)$, where $V$ is a finite set of feature indices.
- Given subset of features $A \subseteq V$, prediction based on $p(y|x_A)$.
Task: pattern recognition based on (at most) features $X_V$ to predict random variable $Y$. True model is $p(Y, X_V)$, where $V$ is a finite set of feature indices.

Given subset of features $A \subseteq V$, prediction based on $p(y|x_A)$.

Goal: choose the smallest set of features that retains accuracy.
Information Gain and Feature Selection

- Task: pattern recognition based on (at most) features $X_V$ to predict random variable $Y$. True model is $p(Y, X_V)$, where $V$ is a finite set of feature indices.
- Given subset of features $A \subseteq V$, prediction based on $p(y|x_A)$.
- Goal: choose the smallest set of features that retains accuracy.
- Information gain is defined as:

$$f(A) = I(Y; X_A) = H(Y) - H(Y|X_A) = H(X_A) - H(X_A|Y) \quad (1.6)$$
Information Gain and Feature Selection

- Task: pattern recognition based on (at most) features $X_V$ to predict random variable $Y$. True model is $p(Y, X_V)$, where $V$ is a finite set of feature indices.
- Given subset of features $A \subseteq V$, prediction based on $p(y|x_A)$.
- Goal: choose the smallest set of features that retains accuracy.
- Information gain is defined as:

$$f(A) = I(Y; X_A) = H(Y) - H(Y|X_A) = H(X_A) - H(X_A|Y)$$ (1.6)

- Goal is to find a subset $A$ of size $k$ that has high information gain.
Information Gain and Feature Selection

- Task: pattern recognition based on (at most) features $X_V$ to predict random variable $Y$. True model is $p(Y, X_V)$, where $V$ is a finite set of feature indices.
- Given subset of features $A \subseteq V$, prediction based on $p(y|x_A)$.
- Goal: choose the smallest set of features that retains accuracy.
- Information gain is defined as:

$$f(A) = I(Y; X_A) = H(Y) - H(Y|X_A) = H(X_A) - H(X_A|Y) \quad (1.6)$$

- Goal is to find a subset $A$ of size $k$ that has high information gain.
- Applicable not only in pattern recognition, but in the sensor coverage problem as well, where $Y$ is whatever question we wish to ask about the room.
Information Theory: Block Coding

Given a set of random variables \( \{X_i\}_{i \in V} \) indexed by set \( V \), how do we partition them so that we can best block-code them within each block.
Information Theory: Block Coding

- Given a set of random variables \( \{X_i\}_{i \in V} \) indexed by set \( V \), how do we partition them so that we can best block-code them within each block.

- I.e., how do we form \( S \subseteq V \) such that \( I(X_S; X_{V \setminus S}) \) is as small as possible, where \( I(X_A; X_B) \) is the mutual information between random variables \( X_A \) and \( X_B \), i.e.,

\[
I(X_A; X_B) = H(X_A) + H(X_B) - H(X_A, X_B) \tag{1.7}
\]

and \( H(X_A) = - \sum_{x_A} p(x_A) \log p(x_A) \) is the joint entropy of the set \( X_A \) of random variables.
A network of senders/receivers
Each sender $X_i$ is trying to communicate simultaneously with each receiver $Y_i$ (i.e., for all $i$, $X_i$ is sending to $\{Y_i\}_i$)
The $X_i$ are not necessarily independent.

Communication rates from $i$ to $j$ are $R^{(i \rightarrow j)}$ to send message $W^{(i \rightarrow j)} \in \{1, 2, \ldots, 2^{nR^{(i \rightarrow j)}}\}$.

Goal: necessary and sufficient conditions for achievability as we’ve done for other channels.
I.e., can we find functions $f$ such that any rates must satisfy
\[
\forall S \subseteq V, \quad \sum_{i \in S, j \in V \setminus S} R^{(i \rightarrow j)} \leq f(S) \quad (1.8)
\]
Monge Matrices

- $m \times n$ matrices $C = [c_{ij}]_{ij}$ are called Monge matrices if they satisfy the Monge property, namely:

$$c_{ij} + c_{rs} \leq c_{is} + c_{rj}$$

(1.9)

for all $1 \leq i < r \leq m$ and $1 \leq j < s \leq n$. 

Useful for speeding up certain dynamic programming problems.
**Monge Matrices**

- $m \times n$ matrices $C = [c_{ij}]_{ij}$ are called Monge matrices if they satisfy the Monge property, namely:

  \[ c_{ij} + c_{rs} \leq c_{is} + c_{rj} \]  

  (1.9)

  for all $1 \leq i < r \leq m$ and $1 \leq j < s \leq n$.

- Consider four elements of the matrix:

\[
\begin{align*}
C_{is} & \quad C_{rs} \\
C_{ij} & \quad C_{rj}
\end{align*}
\]  

[Diagram of elements in a matrix with inequality relationship]
Monge Matrices

- \( m \times n \) matrices \( C = [c_{ij}]_{ij} \) are called Monge matrices if they satisfy the Monge property, namely:

\[
c_{ij} + c_{rs} \leq c_{is} + c_{rj}
\]  

(1.9)

for all \( 1 \leq i < r \leq m \) and \( 1 \leq j < s \leq n \).

- Consider four elements of the matrix:

Useful for speeding up certain dynamic programming problems.
Monge Matrices

Can generate a Monge matrix from a convex polygon - delete two segments, then separately number vertices on each chain. Distances $c_{ij}$ satisfy Monge property (or quadrangle inequality).
Monge Matrices

- Can generate a Monge matrix from a convex polygon - delete two segments, then separately number vertices on each chain. Distances $c_{ij}$ satisfy Monge property (or quadrangle inequality).
Can generate a Monge matrix from a convex polygon - delete two segments, then separately number vertices on each chain. Distances $c_{ij}$ satisfy Monge property (or quadrangle inequality).
Can generate a Monge matrix from a convex polygon - delete two segments, then separately number vertices on each chain. Distances \( c_{ij} \) satisfy Monge property (or quadrangle inequality).
Monge Matrices

- Can generate a Monge matrix from a convex polygon - delete two segments, then separately number vertices on each chain. Distances $c_{ij}$ satisfy Monge property (or quadrangle inequality).
A model of Influence in Social Networks

- Given a graph $G = (V, E)$, each $v \in V$ corresponds to a person, to each $v$ we have an activation function $f_v : 2^V \rightarrow [0, 1]$ dependent only on its neighbors. I.e., $f_v(A) = f_v(A \cap \Gamma(v))$. 
A model of Influence in Social Networks

- Given a graph $G = (V, E)$, each $v \in V$ corresponds to a person, to each $v$ we have an activation function $f_v : 2^V \rightarrow [0, 1]$ dependent only on its neighbors. I.e., $f_v(A) = f_v(A \cap \Gamma(v))$.

- Goal - Viral Marketing: find a small subset $S \subseteq V$ of individuals to directly influence, and thus indirectly influence the greatest number of possible other individuals (via the social network $G$).
A model of Influence in Social Networks

- Given a graph $G = (V, E)$, each $v \in V$ corresponds to a person, to each $v$ we have an activation function $f_v : 2^V \rightarrow [0, 1]$ dependent only on its neighbors. I.e., $f_v(A) = f_v(A \cap \Gamma(v))$.

- Goal - Viral Marketing: find a small subset $S \subseteq V$ of individuals to directly influence, and thus indirectly influence the greatest number of possible other individuals (via the social network $G$).

- We define a function $f : 2^V \rightarrow \mathbb{Z}^+$ that models the ultimate influence of an initial set $S$ of nodes based on the following iterative process: At each step, a given set of nodes $S$ are activated, and we activate new nodes $v \in V \setminus S$ if $f_v(S) \geq U[0, 1]$ (where $U[0, 1]$ is a uniform random number between 0 and 1).
Let $V$ be a group of individuals. How valuable to you is a given friend $v \in V$?
The value of a friend

- Let $V$ be a group of individuals. How valuable to you is a given friend $v \in V$?
- It depends on how many friends you have.
Let $V$ be a group of individuals. How valuable to you is a given friend $v \in V$?

It depends on how many friends you have.

Given a group of friends $S \subseteq V$, can you valuate them with a function $f(S)$ and how?
The value of a friend

- Let $V$ be a group of individuals. How valuable to you is a given friend $v \in V$?
- It depends on how many friends you have.
- Given a group of friends $S \subseteq V$, can you valuate them with a function $f(S)$ and how?
- Let $f(S)$ be the value of the set of friends $S$. Is submodular or supermodular a good model?
Information Cascades, Diffusion Networks

How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.
How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).

How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?

There might be only one seed source (shown below) or many.
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.

[Diagram showing a diffusion network with an original event in the center and multiple cascades diverging from it.]
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.
Information Cascades, Diffusion Networks

- How to model flow of information from source to the point it reaches users — information used in its common sense (like news events).
- How to find the most influential sources, the ones that often set off cascades, which are like large “waves” of information flow?
- There might be only one seed source (shown below) or many.
Information propagation: when blogs or news stories break, and creates an information cascade over multiple other blogs/newspapers/magazines.

Viral marketing: What is the pattern of trendsetters that cause an individual to purchase a product?

Epidemiology: who got sick from whom, and what is the network of such links?

How can we infer the connectivity of a network (of memes, purchase decisions, viruses, etc.) based only on diffusion traces (the time that each node is “infected”)? How to find the most likely tree?
Graphical Models: Tree Distributions

- Family of probability distributions $p : \{0, 1\}^V \rightarrow [0, 1]$: 
  \[ p(x) = \frac{1}{Z} \exp(f(x)) \] (1.10)
Graphical Models: Tree Distributions

- Family of probability distributions $p : \{0, 1\}^V \rightarrow [0, 1]$: 
  \[
p(x) = \frac{1}{Z} \exp(f(x))
  \]  
  (1.10)

- Given a graphical model $G = (V, E)$ and a family of probability distributions $p \in \mathcal{F}(G, M)$ that factor w.r.t. that distribution. I.e., 
  $f(x) = \sum_{c \in C} f_c(x_c)$ where $C$ are a set of cliques.
Graphical Models: Tree Distributions

- Family of probability distributions $p : \{0, 1\}^V \rightarrow [0, 1]$:

  $$p(x) = \frac{1}{Z} \exp(f(x)) \quad (1.10)$$

- Given a graphical model $G = (V, E)$ and a family of probability distributions $p \in \mathcal{F}(G, \mathcal{M})$ that factor w.r.t. that distribution. I.e.,

  $$f(x) = \sum_{c \in C} f_c(x_c)$$

  where $C$ are a set of cliques.

- Find the closest distribution $p_t$ to $p$ subject to $p_t$ factoring w.r.t. some tree $T = (V, F)$, i.e., $p_t \in \mathcal{F}(T, \mathcal{M})$. 

**Optimization Problem**

$$\minimize_{p_t \in \mathcal{F}(G, \mathcal{M})} D(p || p_t) \quad \text{subject to} \quad p_t \in \mathcal{F}(T, \mathcal{M})$$

$T = (V, F)$ is a tree (1.11)

Discrete problem: Choose the right subset of edges from $E$ that make up a tree (i.e., find a spanning tree of $G$ of best quality).

Prof. Jeff Bilmes
Graphical Models: Tree Distributions

- Family of probability distributions \( p : \{0, 1\}^V \to [0, 1] \):
  \[
  p(x) = \frac{1}{Z} \exp(f(x)) \quad (1.10)
  \]

- Given a graphical model \( G = (V, E) \) and a family of probability distributions \( p \in \mathcal{F}(G, \mathcal{M}) \) that factor w.r.t. that distribution. I.e., \( f(x) = \sum_{c \in C} f_c(x_c) \) where \( C \) are a set of cliques.

- Find the closest distribution \( p_t \) to \( p \) subject to \( p_t \) factoring w.r.t. some tree \( T = (V, F) \), i.e., \( p_t \in \mathcal{F}(T, \mathcal{M}) \).

- I.e., optimization problem
  \[
  \begin{align*}
  \text{minimize} & \quad D(p||p_t) \\
  \text{subject to} & \quad p_t \in \mathcal{F}(T, \mathcal{M}) \quad (1.11) \\
  & \quad T = (V, F) \text{ is a tree}
  \end{align*}
  \]

Prof. Jeff Bilmes
Graphical Models: Tree Distributions

- Family of probability distributions \( p : \{0, 1\}^V \to [0, 1] : \)
  \[
p(x) = \frac{1}{Z} \exp(f(x))
  \]

  Given a graphical model \( G = (V, E) \) and a family of probability distributions \( p \in \mathcal{F}(G, \mathcal{M}) \) that factor w.r.t. that distribution. I.e., 
  \[
f(x) = \sum_{C \in \mathcal{C}} f_C(x_C)
  \]
  where \( \mathcal{C} \) are a set of cliques.

- Find the closest distribution \( p_t \) to \( p \) subject to \( p_t \) factoring w.r.t. some tree \( T = (V, F) \), i.e., \( p_t \in \mathcal{F}(T, \mathcal{M}) \).

- I.e., optimization problem
  \[
  \begin{align*}
  \text{minimize} & \quad D(p \| p_t) \\
  \text{subject to} & \quad p_t \in \mathcal{F}(T, \mathcal{M}).
  \end{align*}
  \]
  \[
  T = (V, F) \text{ is a tree}
  \]

- Discrete problem: Choose the right subset of edges from \( E \) that make up a tree (i.e., find a spanning tree of \( G \) of best quality).
Graphical Models: Image Segmentation

- an image needing to be segmented.
labeled data in the form of some pixels being marked foreground (red) and others being marked background (blue).
Graphical Models: Image Segmentation

- the foreground is removed from the background.
Markov random field

\[ \log p(x) \propto \sum_{v \in V(G)} e_v(x_v) + \sum_{(i,j) \in E(G)} e_{ij}(x_i, x_j) \]  \hspace{1cm} (1.12)

When \( G \) is a 2D grid graph, we have
Markov random fields and image segmentation

- We can create auxiliary graph that involves two new nodes s and t and connect each of s and t to all of the original nodes.
- I.e., $G_a = (V \cup \{s, t\}, E + \bigcup_{v \in V} ((s, v) \cup (v, t)))$. 
Markov random fields and image segmentation

Original Graph: \( \log p(x) \propto \sum_{v \in V(G)} e_v(x_v) + \sum_{(i,j) \in E(G)} e_{ij}(x_i, x_j) \)
Augmented graph-cut graph. The edge weights of graph are derived from \( \{e_v\}_{v \in V} \) and \( \{e_{ij}\}_{(i,j) \in E(G)} \)
Markov random fields and image segmentation

Augmented graph-cut graph with indicated cut corresponding to particular vector $\bar{x} \in \{0, 1\}^n$. Each cut $\bar{x}$ has a score corresponding to $p(\bar{x})$. 
Other applications in or related to computer vision

- Image denoising, total variation, structured convex norms.

\[ g(w) = \sum_{i=2}^{N} |w_i - w_{i-1}| \]  

(from Rodriguez, 2009)

- Multi-label graph cuts
- Graphical model inference, computing the Viterbi (or the MPE or the MAP) assignment of a set of random variables.
- Clustering of data sets.
Diverse web search. Given search term (e.g., “jaguar”) but no other information, one probably does not want only articles about cars.
Diverse web search. Given search term (e.g., “jaguar”) but no other information, one probably does not want only articles about cars.

Given a set $V$ of items, how do we choose a subset $S \subseteq V$ that is as diverse as possible, with perhaps constraints on $S$ such as its size.
Diverse web search. Given search term (e.g., “jaguar”) but no other information, one probably does not want only articles about cars.

Given a set $V$ of items, how do we choose a subset $S \subseteq V$ that is as diverse as possible, with perhaps constraints on $S$ such as its size.

How do we choose the smallest set $S$ that maintains a given quality of diversity?
Diverse web search. Given search term (e.g., “jaguar”) but no other information, one probably does not want only articles about cars.

Given a set $V$ of items, how do we choose a subset $S \subseteq V$ that is as diverse as possible, with perhaps constraints on $S$ such as its size.

How do we choose the smallest set $S$ that maintains a given quality of diversity?

Goal of diversity: ensure proper representation in chosen set that, say otherwise in a random sample, could lead to poor representation of normally underrepresented groups.
Extractive Document Summarization

- The figure below represents the sentences of a document
We extract sentences (green) as a summary of the full document.
Extractive Document Summarization

- We extract sentences (green) as a summary of the full document:
  
  
  
  
  
  

Extractive Document Summarization

- We extract sentences (green) as a summary of the full document

\[ \subset \]

- The summary on the left is a subset of the summary on the right.

Consider adding a new (blue) sentence to each of the two summaries. The marginal (incremental) benefit of adding the new (blue) sentence to the smaller (left) summary is no more than the marginal benefit of adding the new sentence to the larger (right) summary.
Extractive Document Summarization

- We extract sentences (green) as a summary of the full document

- The summary on the left is a subset of the summary on the right.
- Consider adding a new (blue) sentence to each of the two summaries.
Extractive Document Summarization

- We extract sentences (green) as a summary of the full document

- The summary on the left is a subset of the summary on the right.

- Consider adding a new (blue) sentence to each of the two summaries.

- The marginal (incremental) benefit of adding the new (blue) sentence to the smaller (left) summary is no more than the marginal benefit of adding the new sentence to the larger (right) summary.
Extractive Document Summarization

- We extract sentences (green) as a summary of the full document.

\[
\text{left summary} \subseteq \text{right summary}
\]

- The summary on the left is a subset of the summary on the right.
- Consider adding a new (blue) sentence to each of the two summaries.
- The marginal (incremental) benefit of adding the new (blue) sentence to the smaller (left) summary is no more than the marginal benefit of adding the new sentence to the larger (right) summary.
- **diminishing returns \(\leftrightarrow\) submodularity**
Web search and information retrieval

- A web search is a form of summarization based on query.
- Goal of a web search engine is to produce a ranked list of web pages that, conditioned on the text query entered, summarizes the most important links on the web.
- Information retrieval (the science of automatically acquiring information), book and music recommendation systems —
- Overall goal: user should quickly find information that is informative, concise, accurate, relevant (to the user’s needs), and comprehensive.
Definition: Set functions

Motivation & Applications

Active Learning and Semi-Supervised Learning

- Given training data $\mathcal{D}_V = \{(x_i, y_i)\}_{i \in V}$ of $(x, y)$ pairs where $x$ is a query (data item) and $y$ is an answer (label), goal is to learn a good mapping $y = h(x)$.
Active Learning and Semi-Supervised Learning

- Given training data $\mathcal{D}_V = \{(x_i, y_i)\}_{i \in V}$ of $(x, y)$ pairs where $x$ is a query (data item) and $y$ is an answer (label), goal is to learn a good mapping $y = h(x)$.

- Often, getting $y$ is time-consuming, expensive, and error prone (manual transcription, Amazon Turk, etc.)
Active Learning and Semi-Supervised Learning

- Given training data $\mathcal{D}_V = \{(x_i, y_i)\}_{i \in V}$ of $(x, y)$ pairs where $x$ is a query (data item) and $y$ is an answer (label), goal is to learn a good mapping $y = h(x)$.

- Often, getting $y$ is time-consuming, expensive, and error prone (manual transcription, Amazon Turk, etc.)

- Batch active learning: choose a subset $S \subset V$ so that only the labels $\{y_i\}_{i \in S}$ should be acquired.
Active Learning and Semi-Supervised Learning

- Given training data $\mathcal{D}_V = \{(x_i, y_i)\}_{i \in V}$ of $(x, y)$ pairs where $x$ is a query (data item) and $y$ is an answer (label), goal is to learn a good mapping $y = h(x)$.

- Often, getting $y$ is time-consuming, expensive, and error prone (manual transcription, Amazon Turk, etc.)

- Batch active learning: choose a subset $S \subset V$ so that only the labels $\{y_i\}_{i \in S}$ should be acquired.

- Adaptive active learning: choose a policy whereby we choose an $i_1 \in V$, get the label $y_{i_1}$, choose another $i_2 \in V$, get label $y_{i_2}$, where each chose can be based on previously acquired labels.
Active Learning and Semi-Supervised Learning

- Given training data $\mathcal{D}_V = \{ (x_i, y_i) \}_{i \in V}$ of $(x, y)$ pairs where $x$ is a query (data item) and $y$ is an answer (label), goal is to learn a good mapping $y = h(x)$.

- Often, getting $y$ is time-consuming, expensive, and error prone (manual transcription, Amazon Turk, etc.)

- Batch active learning: choose a subset $S \subset V$ so that only the labels $\{y_i\}_{i \in S}$ should be acquired.

- Adaptive active learning: choose a policy whereby we choose an $i_1 \in V$, get the label $y_{i_1}$, choose another $i_2 \in V$, get label $y_{i_2}$, where each chose can be based on previously acquired labels.

- Semi-supervised (transductive) learning: Once we have $\{y_i\}_{i \in S}$, infer the remaining labels $\{y_i\}_{i \in V \setminus S}$.