Graphs in Machine Learning

Daniele Calandriello

DeepMind Paris, France

Collaborators: Achraf Azize,
Michal Valko

Partially based on material by: Andreas Krause,
Branislav Kveton, Michael Kearns

Administrivia

7/8 lectures 2x(50m + 10m break)

3 recitations (TDs) 2h + 2 weeks to deliver homework

TA: Achraf Azize

Administrivia

7/8 lectures 2x(50m + 10m break)

3 recitations (TDs) 2h + 2 weeks to deliver homework

TA: Achraf Azize

Time: Mondays 16:00-18:00, 23/1 to 3/4

Place: online, link posted on Piazza

Administrivia

7/8 lectures 2x(50m + 10m break)

3 recitations (TDs) 2h + 2 weeks to deliver homework

TA: Achraf Azize

Time: Mondays 16:00-18:00, 23/1 to 3/4

Place: online, link posted on Piazza

Validation: grades from TDs

Piazza for Q&A's

Contact, online class discussions, and announcements:

- registration for the class
- register with your **school** email and **full name**
- online course discussions and announcements
- questions and answers about the material and logistics
- **students encouraged to answer each others' questions**
- homework assignments
- draft of the slides before the class

`https://piazza.com/ens_cachan/spring2023/mvagraphsml` **NO EMAILS!**

class code: **mvagraphsml** write it down right now + register with your school email

Course website:

`https://sites.google.com/view/daniele-calandriello/teaching`

Two (main) sources of graphs in ML

Natural graphs as models for networks

Constructed graphs as nonparametric basis

Two (main) sources of graphs in ML

Natural graphs as models for networks

- given as an input

Constructed graphs as nonparametric basis

Two (main) sources of graphs in ML

Natural graphs as models for networks

- given as an input
- discover interesting properties of the structure

Constructed graphs as nonparametric basis

Two (main) sources of graphs in ML

Natural graphs as models for networks

- given as an input
- discover interesting properties of the structure

Constructed graphs as nonparametric basis

- we create (learn) the similarity structure from flat data

Two (main) sources of graphs in ML

Natural graphs as models for networks

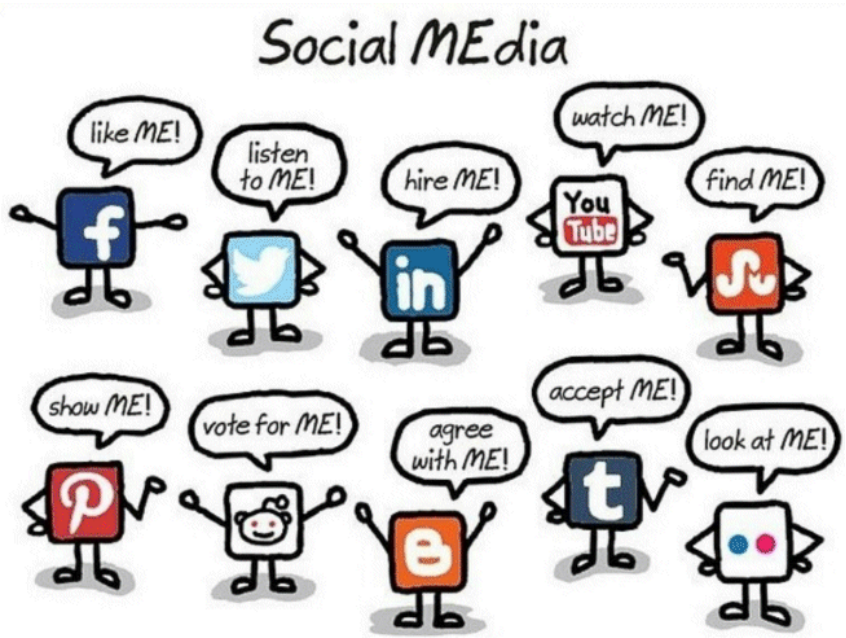
- given as an input
- discover interesting properties of the structure

Constructed graphs as nonparametric basis

- we create (learn) the similarity structure from flat data
- it's a tool (e.g., nonparametric regularizer) to encode structural properties (e.g., independence, . . .)

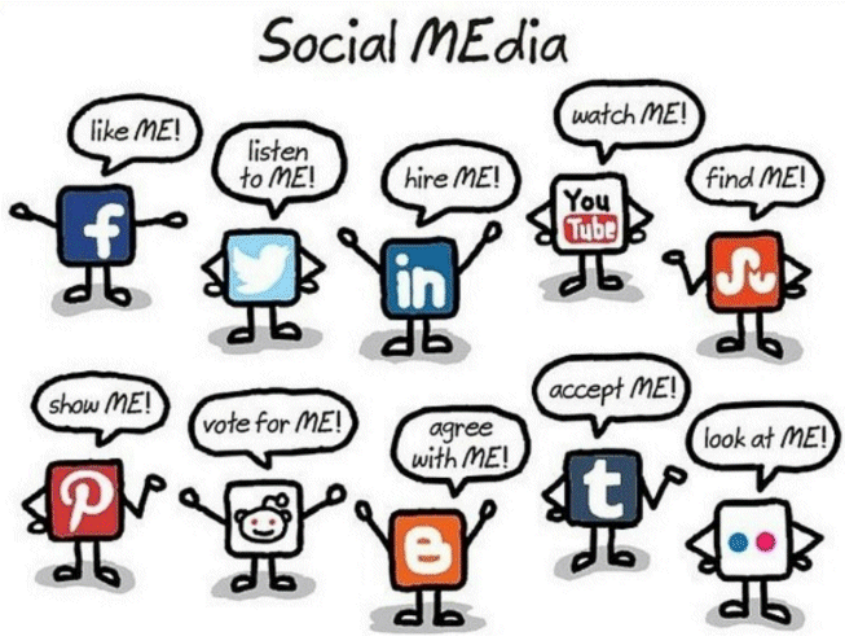
Natural graphs from social networks

- people and their interactions



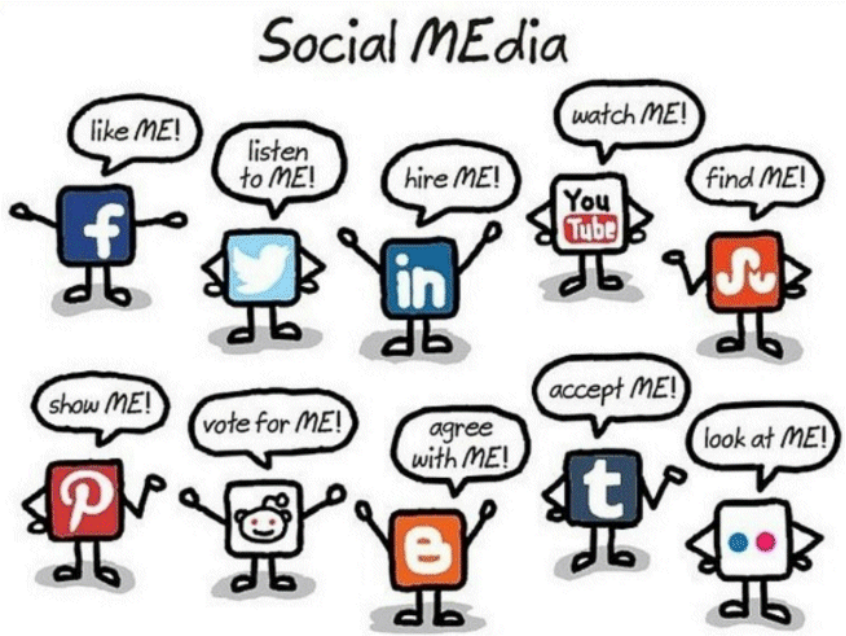
Natural graphs from social networks

- people and their interactions
- structure is rather a *phenomena*



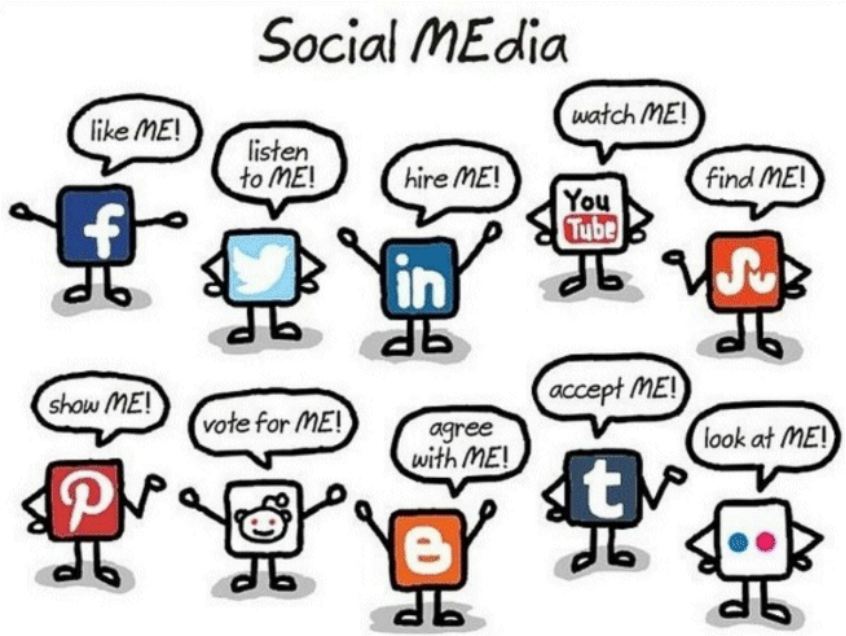
Natural graphs from social networks

- people and their interactions
- structure is rather a *phenomena*
- typical ML tasks



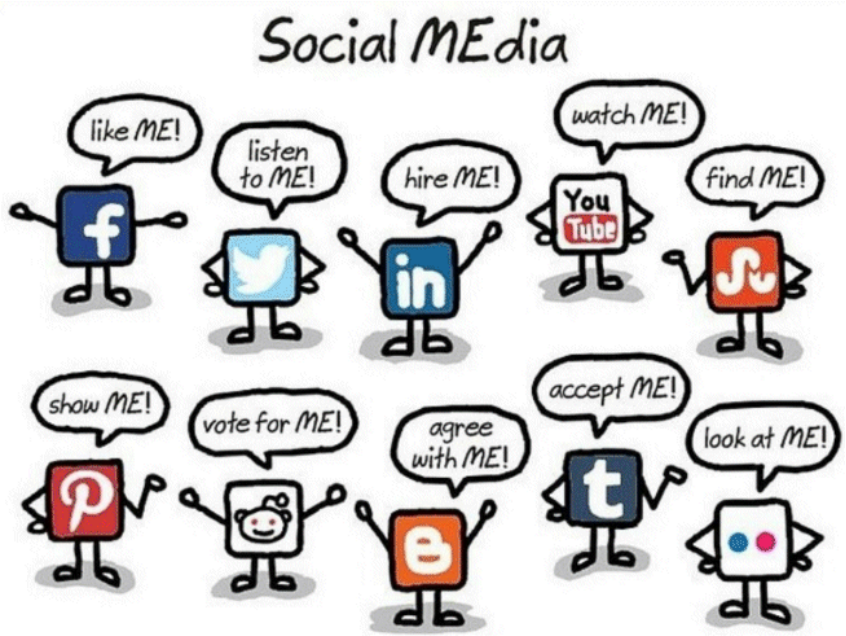
Natural graphs from social networks

- people and their interactions
- structure is rather a *phenomena*
- typical ML tasks
  - advertising



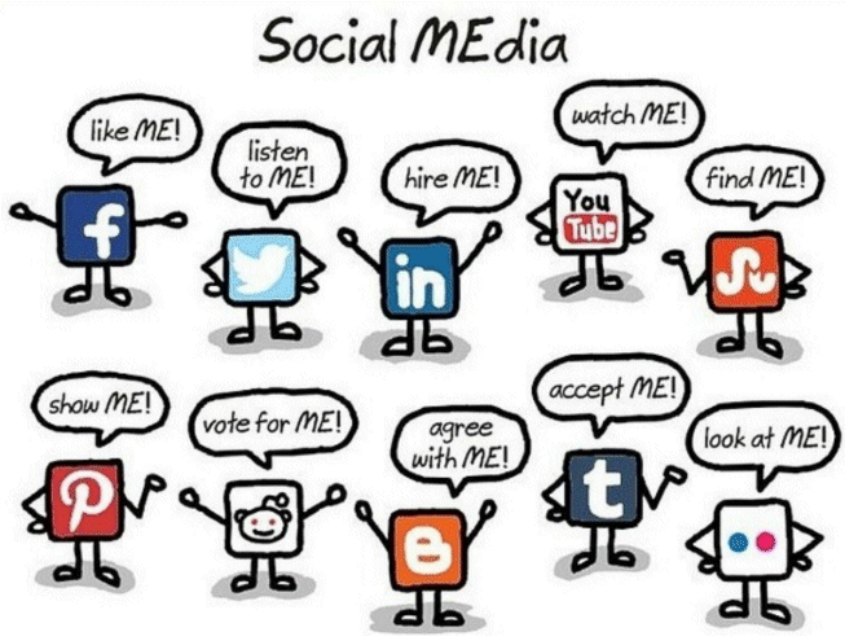
Natural graphs from social networks

- people and their interactions
- structure is rather a *phenomena*
- typical ML tasks
  - advertising
  - link prediction (PYMK)

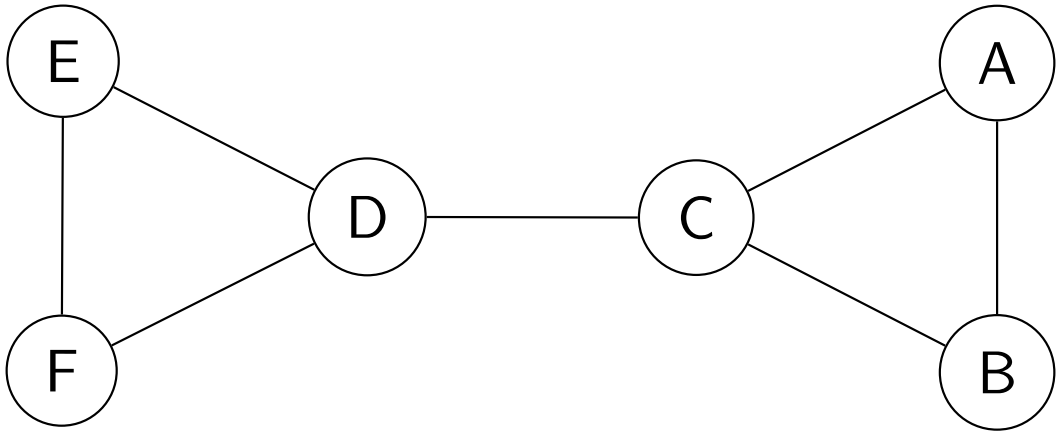


Natural graphs from social networks

- people and their interactions
- structure is rather a *phenomena*
- typical ML tasks
  - advertising
  - link prediction (PYMK)
  - find influential sources

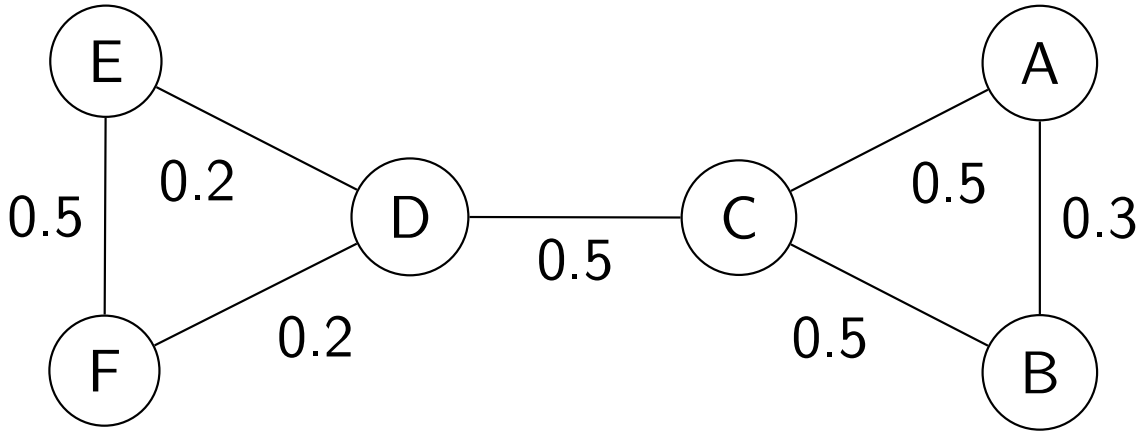


Success story #1 Product placement - problem



Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem

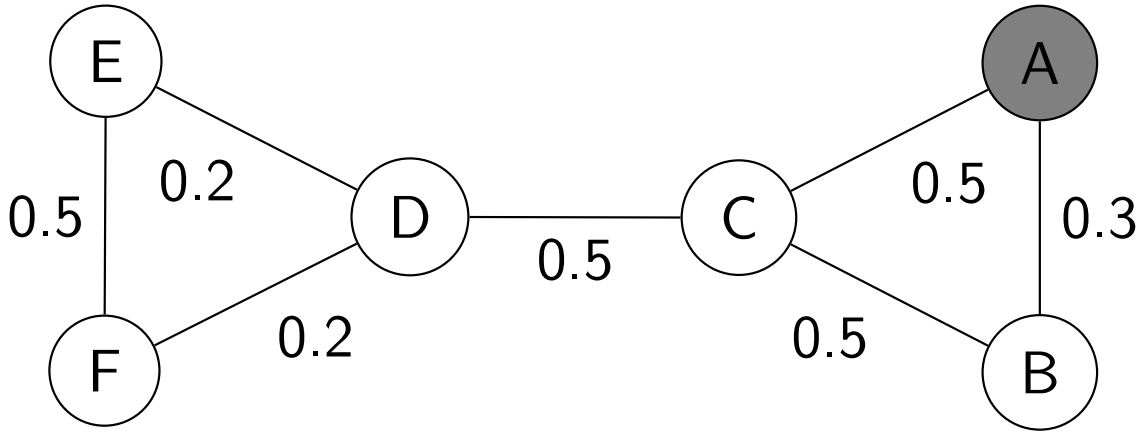


Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem

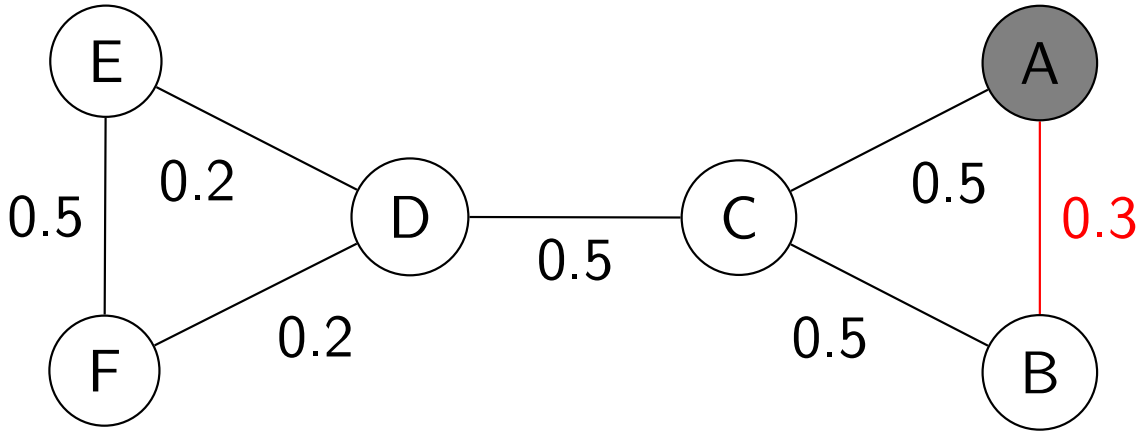


Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem

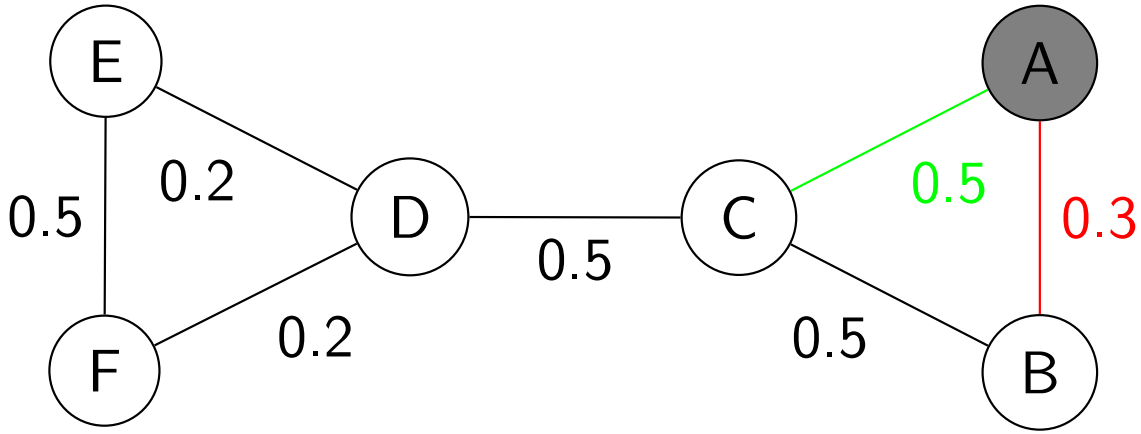


Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem

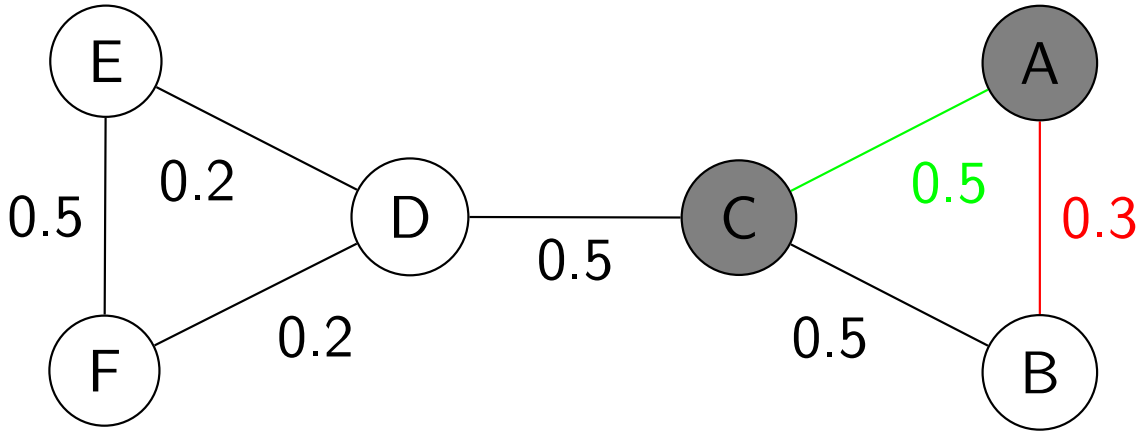


Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem

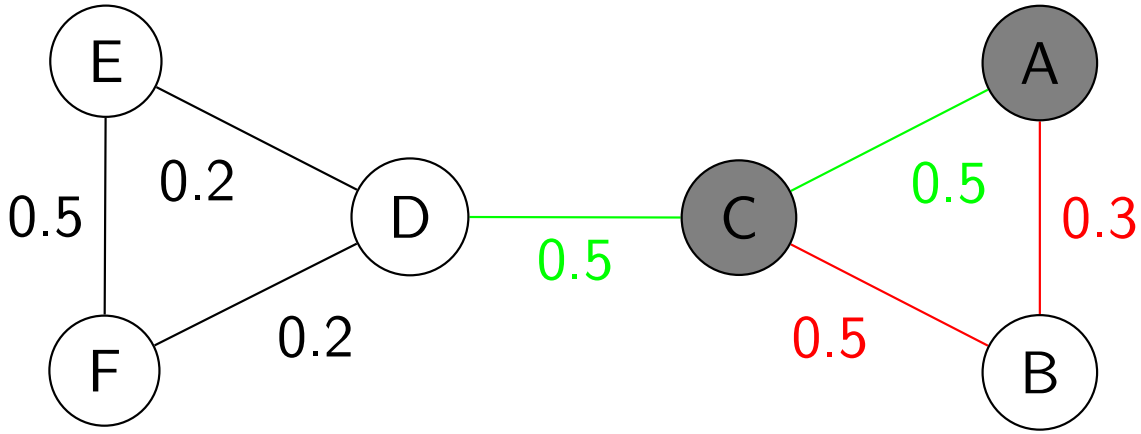


Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem

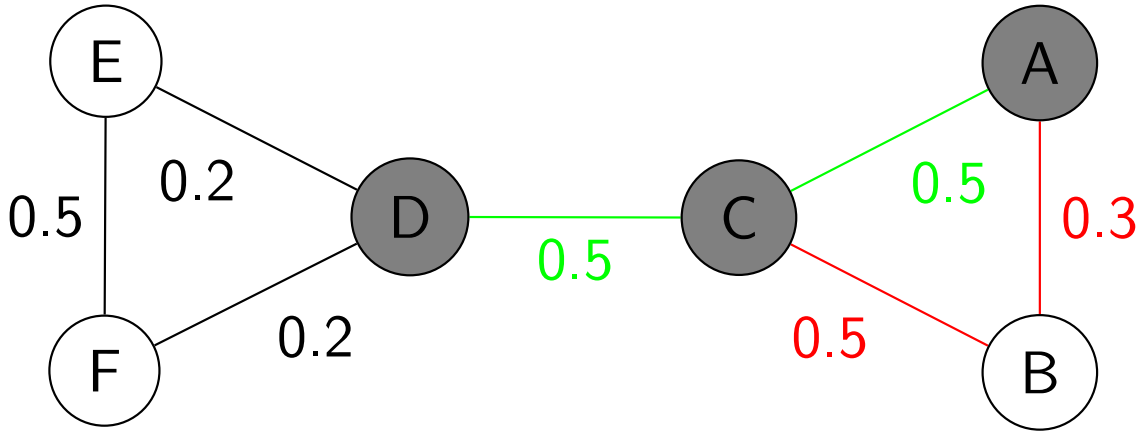


Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem

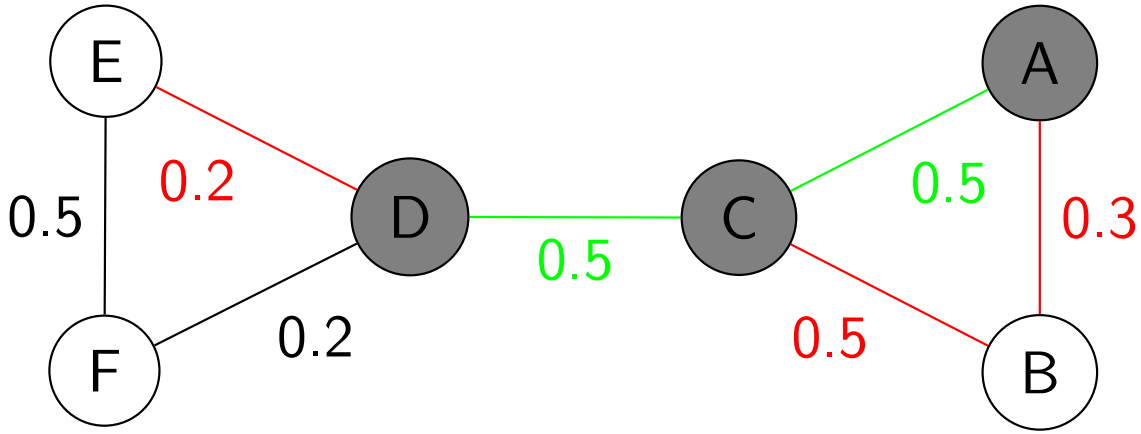


Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem

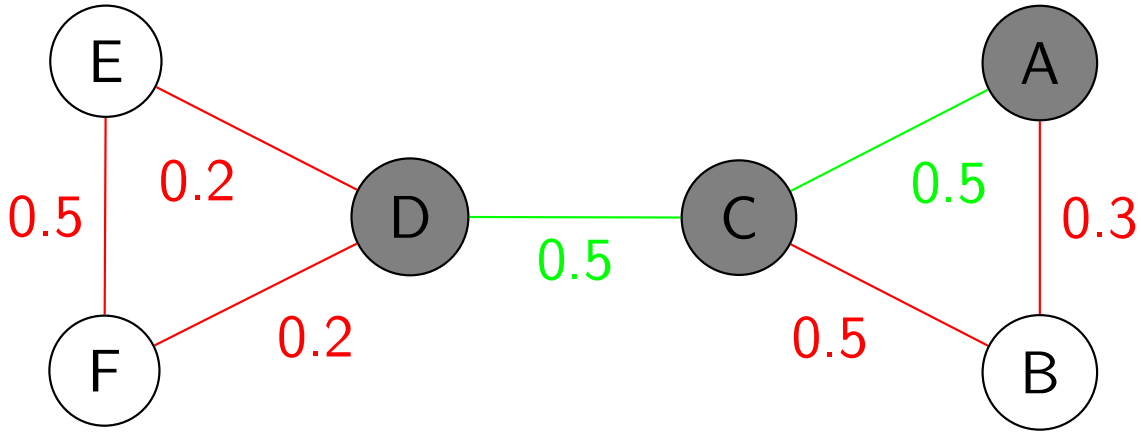


Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem



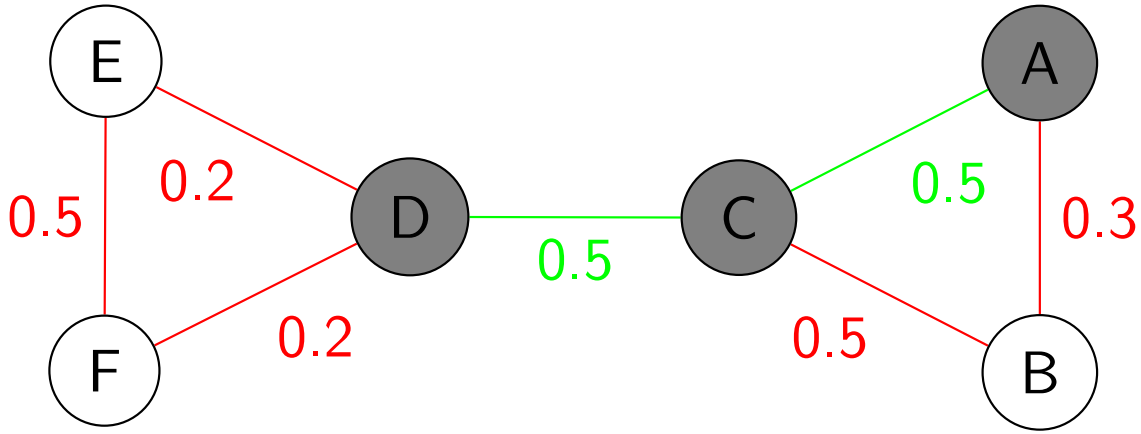
Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

$F(S)$ = Expected number of people influenced when targeting $S \subseteq V$ under some propagation model - e.g., cascades

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem



Who should get free cell phones?

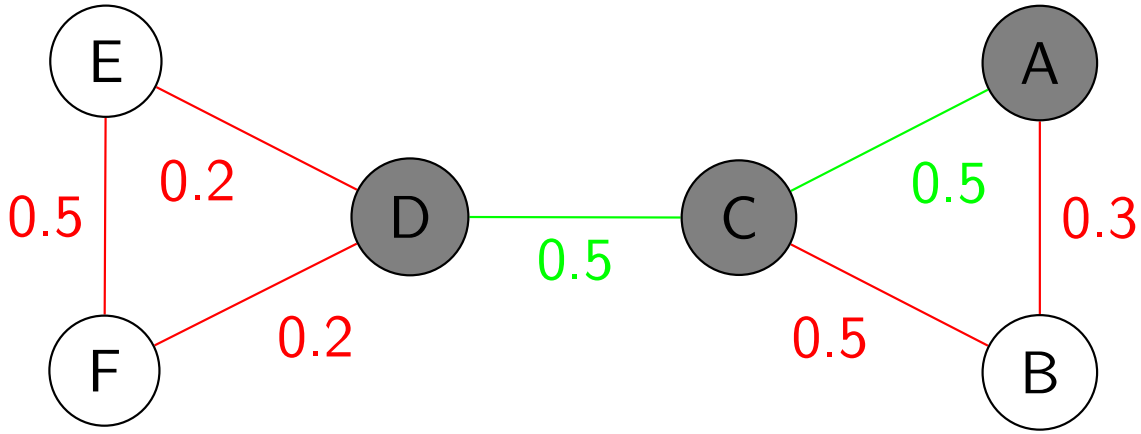
$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

$F(S)$ = Expected number of people influenced when targeting $S \subseteq V$ under some propagation model - e.g., cascades

How would you choose the target customers?

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Success story #1 Product placement - problem



Who should get free cell phones?

$V = \{$**A**lice,**B**ob,**C**harlie,**D**orothy,**E**ric,**F**iona$\}$

$F(S)$ = Expected number of people influenced when targeting $S \subseteq V$ under some propagation model - e.g., cascades

How would you choose the target customers?

highest degree, close to the center, . . .

Maximizing the Spread of Influence through a Social Network
`http://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf`

Submodularity: modeling diminishing returns

Example: $S = \{\text{stuff}\} = \{\text{bread, apple, tomato, ...}\}$

$f(V) =$ cost of getting products V

Submodularity: modeling diminishing returns

Example: $S = \{\text{stuff}\} = \{\text{bread, apple, tomato, ...}\}$

$f(V) =$ cost of getting products V

$f(\{\text{bread}\}) = c(\text{bakery}) + c(\text{bread})$

$f(\{\text{bread, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{apple}\})$

$f(\{\text{bread, tomato}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato})$

$f(\{\text{bread, tomato, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato}) + c(\text{apple})$

Submodularity: modeling diminishing returns

Example: $S = \{\text{stuff}\} = \{\text{bread, apple, tomato, ...}\}$

$f(V) =$ cost of getting products V

$f(\{\text{bread}\}) = c(\text{bakery}) + c(\text{bread})$

$f(\{\text{bread, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{apple}\})$

$f(\{\text{bread, tomato}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato})$

$f(\{\text{bread, tomato, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato}) + c(\text{apple})$

Adding an apple to the smaller set costs more!

Submodularity: modeling diminishing returns

Example: $S = \{\text{stuff}\} = \{\text{bread, apple, tomato, ...}\}$

$f(V) =$ cost of getting products V

$f(\{\text{bread}\}) = c(\text{bakery}) + c(\text{bread})$

$f(\{\text{bread, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{apple}\})$

$f(\{\text{bread, tomato}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato})$

$f(\{\text{bread, tomato, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato}) + c(\text{apple})$

Adding an apple to the smaller set costs more!

$$\{\text{bread}\} \subseteq \{\text{bread, tomato}\}$$

$$f(\{\text{bread, apple}\}) - f(\{\text{bread}\}) > f(\{\text{bread, tomato, apple}\}) - f(\{\text{tomato, bread}\})$$

Submodularity: modeling diminishing returns

Example: $S = \{\text{stuff}\} = \{\text{bread, apple, tomato, ...}\}$

$f(V) =$ cost of getting products V

$f(\{\text{bread}\}) = c(\text{bakery}) + c(\text{bread})$

$f(\{\text{bread, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{apple}\})$

$f(\{\text{bread, tomato}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato})$

$f(\{\text{bread, tomato, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato}) + c(\text{apple})$

Adding an apple to the smaller set costs more!

$$\{\text{bread}\} \subseteq \{\text{bread, tomato}\}$$

$$f(\{\text{bread, apple}\}) - f(\{\text{bread}\}) > f(\{\text{bread, tomato, apple}\}) - f(\{\text{tomato, bread}\})$$

Diminishing returns: Buying in bulk is cheaper!

Submodularity: modeling diminishing returns

Example: $S = \{\text{stuff}\} = \{\text{bread, apple, tomato, ...}\}$

$f(V) =$ cost of getting products V

$f(\{\text{bread}\}) = c(\text{bakery}) + c(\text{bread})$

$f(\{\text{bread, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{apple}\})$

$f(\{\text{bread, tomato}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato})$

$f(\{\text{bread, tomato, apple}\}) = c(\text{bakery}) + c(\text{bread}) + c(\text{market}) + c(\text{tomato}) + c(\text{apple})$

Adding an apple to the smaller set costs more!

$$\{\text{bread}\} \subseteq \{\text{bread, tomato}\}$$

$$f(\{\text{bread, apple}\}) - f(\{\text{bread}\}) > f(\{\text{bread, tomato, apple}\}) - f(\{\text{tomato, bread}\})$$

Diminishing returns: Buying in bulk is cheaper!

A **set function** on a discrete set A is **submodular** if for any $S \subseteq T \subseteq A$ and for any $e \in A \setminus T$

$$f(S \cup \{e\}) - f(S) \geq f(T \cup \{e\}) - f(T)$$

Submodularity: Application

Link to our **product placement** problem on a **social network graph?**

Objective: Find $\arg\max_{S \subseteq A, |S| \leq k} f(S)$

Submodularity: Application

Link to our **product placement** problem on a **social network graph?**

Objective: Find $\arg\max_{S \subseteq A, |S| \leq k} f(S)$

Property: NP-hard in general

Special case: f is **nonnegative**, **submodular** and **monotone**.

Submodularity: Application

Link to our **product placement** problem on a **social network graph?**

Objective: Find $\arg\max_{S \subseteq A, |S| \leq k} f(S)$

Property: NP-hard in general

Special case: f is **nonnegative**, **submodular** and **monotone**.

`http://thibaut.horel.org/submodularity/papers/nemhauser1978.pdf`

Let $S^\star = \arg\max_{S \subseteq A, |S| \leq k} f(S)$ where f is monotonic and submodular set function and let $S_{\texttt{Greedy}}$ be a **greedy solution**.

$$\text{Then} \quad f(S_{\texttt{Greedy}}) \geq \left(1 - \frac{1}{e}\right) \cdot f(S^\star).$$

Submodularity: Application

Link to our **product placement** problem on a **social network graph?**

Objective: Find $\arg\max_{S \subseteq A, |S| \leq k} f(S)$

Property: NP-hard in general

Special case: f is **nonnegative**, **submodular** and **monotone**.

`http://thibaut.horel.org/submodularity/papers/nemhauser1978.pdf`

Let $S^\star = \arg\max_{S \subseteq A, |S| \leq k} f(S)$ where f is monotonic and submodular set function and let $S_{\texttt{Greedy}}$ be a **greedy solution**.

$$\text{Then} \quad f(S_{\texttt{Greedy}}) \geq \left(1 - \tfrac{1}{e}\right) \cdot f(S^\star).$$

Other applications: information, graph cuts, covering, ...

Submodularity: `Greedy` algorithm

Input:
  k: the maximum allowed cardinality of the output
  V: a ground set
  f: a monotone, non-negative, and submodular function
Run:
$S_0 = \emptyset$
for $i = 1$ **to** k **do**
  $S_i \leftarrow S_{i-1} \cup \left\{ \arg\max_{a \in V \setminus S_{i-1}} \left[f\left(\{a\} \cup S_{i-1}\right) - f\left(S_{i-1}\right) \right] \right\}$
end for
Output:
  Return $S_{\texttt{Greedy}} = S_k$

Let $S^\star = \arg\max_{S \subseteq A, |S| \leq k} f(S)$ where f is monotonic and submodular set function and let $S_{\texttt{Greedy}}$ be a **greedy solution**.

$$\text{Then} \quad f(S_{\texttt{Greedy}}) \geq \left(1 - \frac{1}{e}\right) \cdot f(S^\star).$$

Submodularity: Approximation guarantee of `Greedy`

Let S_i be the i-th set selected by `Greedy`. We show

$$f(S^\star) - f(S_{i-1}) \leq f(S^\star \cup S_{i-1}) - f(S_{i-1})$$

Submodularity: Approximation guarantee of `Greedy`

Let S_i be the i-th set selected by `Greedy`. We show

$$f(S^\star) - f(S_{i-1}) \leq f(S^\star \cup S_{i-1}) - f(S_{i-1})$$
$$\leq f(a \cup S_{i-1}) - f(S_{i-1}) + f(S^\star / a \cup S_{i-1}) - f(S_{i-1})$$

Submodularity: Approximation guarantee of `Greedy`

Let S_i be the i-th set selected by `Greedy`. We show

$$
\begin{aligned}
& f\left(S^{\star}\right)-f\left(S_{i-1}\right) \leq f\left(S^{\star} \cup S_{i-1}\right)-f\left(S_{i-1}\right) \\
& \leq f(a \cup S_{i-1})-f(S_{i-1})+f(S^{\star} / a \cup S_{i-1})-f(S_{i-1}) \\
& \leq \sum_{a \in S^{\star} \setminus S_{i-1}}\left(f\left(\{a\} \cup S_{i-1}\right)-f\left(S_{i-1}\right)\right)
\end{aligned}
$$

Submodularity: Approximation guarantee of `Greedy`

Let S_i be the i-th set selected by `Greedy`. We show

$$
\begin{aligned}
& f\left(S^{\star}\right)-f\left(S_{i-1}\right) \leq f\left(S^{\star} \cup S_{i-1}\right)-f\left(S_{i-1}\right) \\
& \leq f(a \cup S_{i-1})-f(S_{i-1})+f(S^{\star}/a \cup S_{i-1})-f(S_{i-1}) \\
& \leq \sum_{a \in S^{\star} \setminus S_{i-1}}\left(f\left(\{a\} \cup S_{i-1}\right)-f\left(S_{i-1}\right)\right) \\
& \leq \sum_{a \in S^{\star} \setminus S_{i-1}}\left(f\left(S_i\right)-f\left(S_{i-1}\right)\right) \leq k\left(f\left(S_i\right)-f\left(S_{i-1}\right)\right)
\end{aligned}
$$

Submodularity: Approximation guarantee of `Greedy`

Let S_i be the i-th set selected by `Greedy`. We show

$$
\begin{aligned}
& f(S^\star) - f(S_{i-1}) \leq f(S^\star \cup S_{i-1}) - f(S_{i-1}) \\
& \leq f(a \cup S_{i-1}) - f(S_{i-1}) + f(S^\star / a \cup S_{i-1}) - f(S_{i-1}) \\
& \leq \sum_{a \in S^\star \setminus S_{i-1}} \left(f(\{a\} \cup S_{i-1}) - f(S_{i-1}) \right) \\
& \leq \sum_{a \in S^\star \setminus S_{i-1}} \left(f(S_i) - f(S_{i-1}) \right) \leq k \left(f(S_i) - f(S_{i-1}) \right)
\end{aligned}
$$

Difference from the optimum of $S_{\texttt{Greedy}} = S_k$ after the k-th step ...

$$
\begin{aligned}
f(S^\star) - f(S_k) &= f(S^\star) - f(S_{k-1}) - \left(f(S_k) - f(S_{k-1}) \right) \\
&\leq f(S^\star) - f(S_{k-1}) - \frac{f(S^\star) - f(S_{k-1})}{k} \\
&\leq \left(1 - \tfrac{1}{k}\right) \cdot \left(f(S^\star) - f(S_{k-1}) \right) \leq \left(1 - \tfrac{1}{k}\right)^k \cdot f(S^\star)
\end{aligned}
$$

Submodularity: Graph-related examples

- Influence maximization on networks (current example)

Submodularity: Graph-related examples

- Influence maximization on networks (current example)
- Maximum-weight spanning trees

Submodularity: Graph-related examples

- Influence maximization on networks (current example)
- Maximum-weight spanning trees
- Graph cuts

Submodularity: Graph-related examples

- Influence maximization on networks (current example)
- Maximum-weight spanning trees
- Graph cuts
- Structure learning in graphical models (PGM course)

Submodularity: Graph-related examples

- Influence maximization on networks (current example)
- Maximum-weight spanning trees
- Graph cuts
- Structure learning in graphical models (PGM course)
- More examples http://people.math.gatech.edu/~tetali/LINKS/IWATA/SFGT.pdf

Submodularity: Graph-related examples

- Influence maximization on networks (current example)
- Maximum-weight spanning trees
- Graph cuts
- Structure learning in graphical models (PGM course)
- More examples http://people.math.gatech.edu/~tetali/LINKS/IWATA/SFGT.pdf
- Deep Submodular Functions (2017) https://arxiv.org/pdf/1701.08939.pdf

Submodularity: Graph-related examples

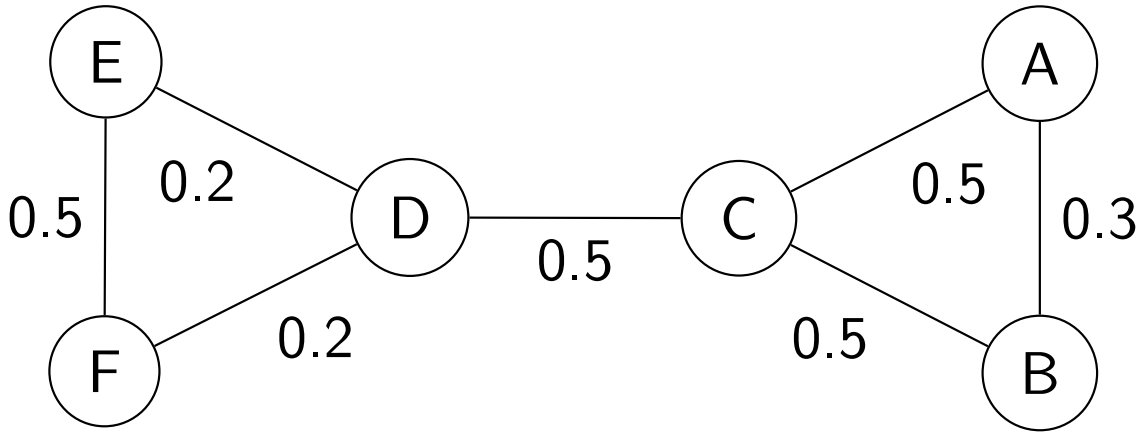
- Influence maximization on networks (current example)
- Maximum-weight spanning trees
- Graph cuts
- Structure learning in graphical models (PGM course)
- More examples http://people.math.gatech.edu/~tetali/LINKS/IWATA/SFGT.pdf
- Deep Submodular Functions (2017) https://arxiv.org/pdf/1701.08939.pdf

Submodularity: Graph-related examples

- Influence maximization on networks (current example)
- Maximum-weight spanning trees
- Graph cuts
- Structure learning in graphical models (PGM course)
- More examples `http://people.math.gatech.edu/~tetali/LINKS/IWATA/SFGT.pdf`
- Deep Submodular Functions (2017) `https://arxiv.org/pdf/1701.08939.pdf`

back to the influence-maximization example ...

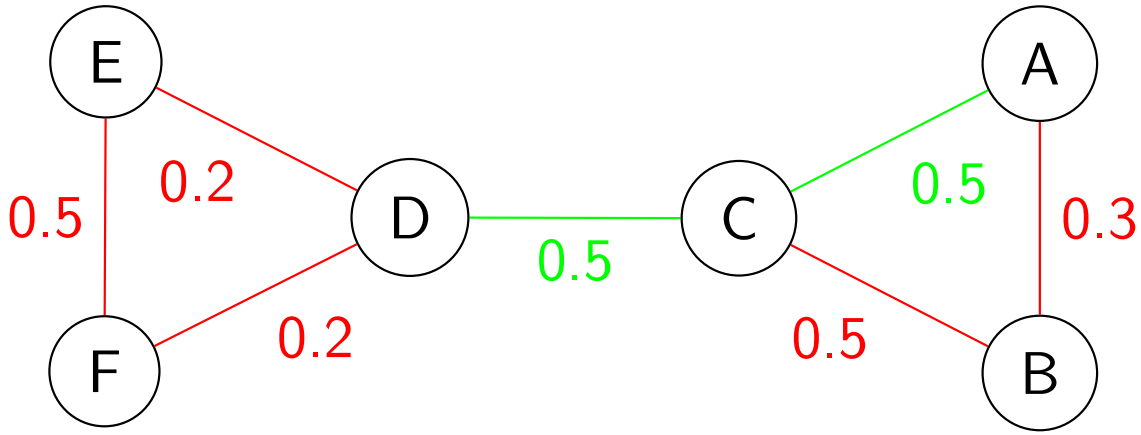
Success story #1 Product placement - solution



Key idea: Flip coins c in advance $\rightarrow$ "live" edges

MIIA: `http://hanj.cs.illinois.edu/pdf/dmkd12_cwang.pdf/`
Tutorial: cf. Andreas Krause `http://submodularity.org/`
Course: Jeff Billmes at UW

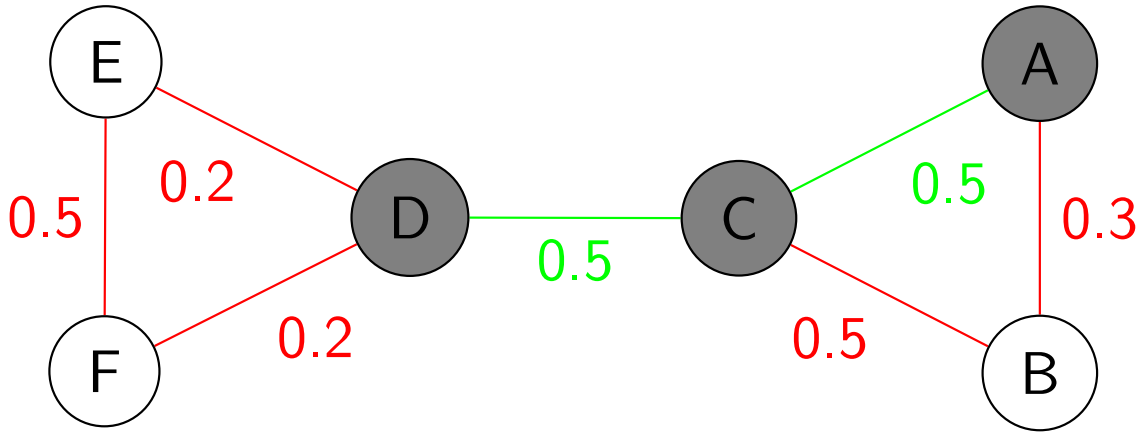
Success story #1 Product placement - solution



Key idea: Flip coins c in advance $\rightarrow$ "live" edges

MIIA: `http://hanj.cs.illinois.edu/pdf/dmkd12_cwang.pdf/`
Tutorial: cf. Andreas Krause `http://submodularity.org/`
Course: Jeff Billmes at UW

Success story #1 Product placement - solution

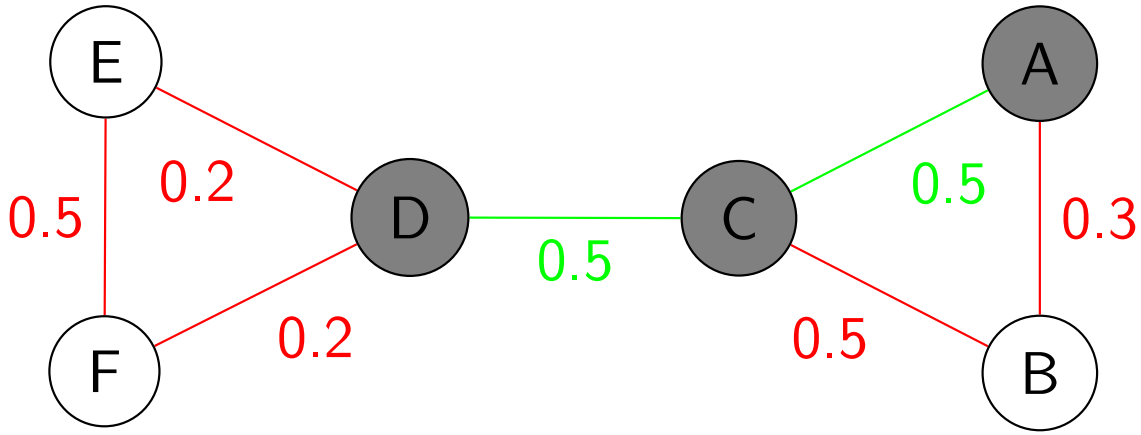


Key idea: Flip coins c in advance $\rightarrow$ "live" edges

$F_c(V)$ = People influenced under outcome c (set cover!)

MIIA: `http://hanj.cs.illinois.edu/pdf/dmkd12_cwang.pdf/`
Tutorial: cf. Andreas Krause `http://submodularity.org/`
Course: Jeff Billmes at UW

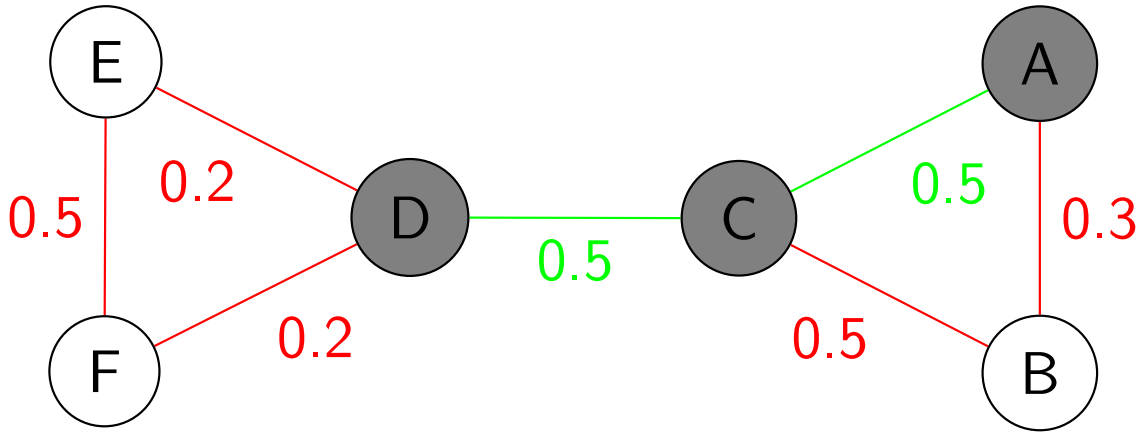
Success story #1 Product placement - solution



Key idea: Flip coins c in advance $\rightarrow$ "live" edges
$F_c(V)$ = People influenced under outcome c (set cover!)
$F(V) = \sum_c P(c)F_c(V)$ is submodular as well!

MIIA: `http://hanj.cs.illinois.edu/pdf/dmkd12_cwang.pdf/`
Tutorial: cf. Andreas Krause `http://submodularity.org/`
Course: Jeff Billmes at UW

Success story #1 Product placement - solution



Key idea: Flip coins c in advance $\rightarrow$ "live" edges

$F_c(V)$ = People influenced under outcome c (set cover!)

$F(V) = \sum_c P(c)F_c(V)$ is submodular as well!

Computational issues?

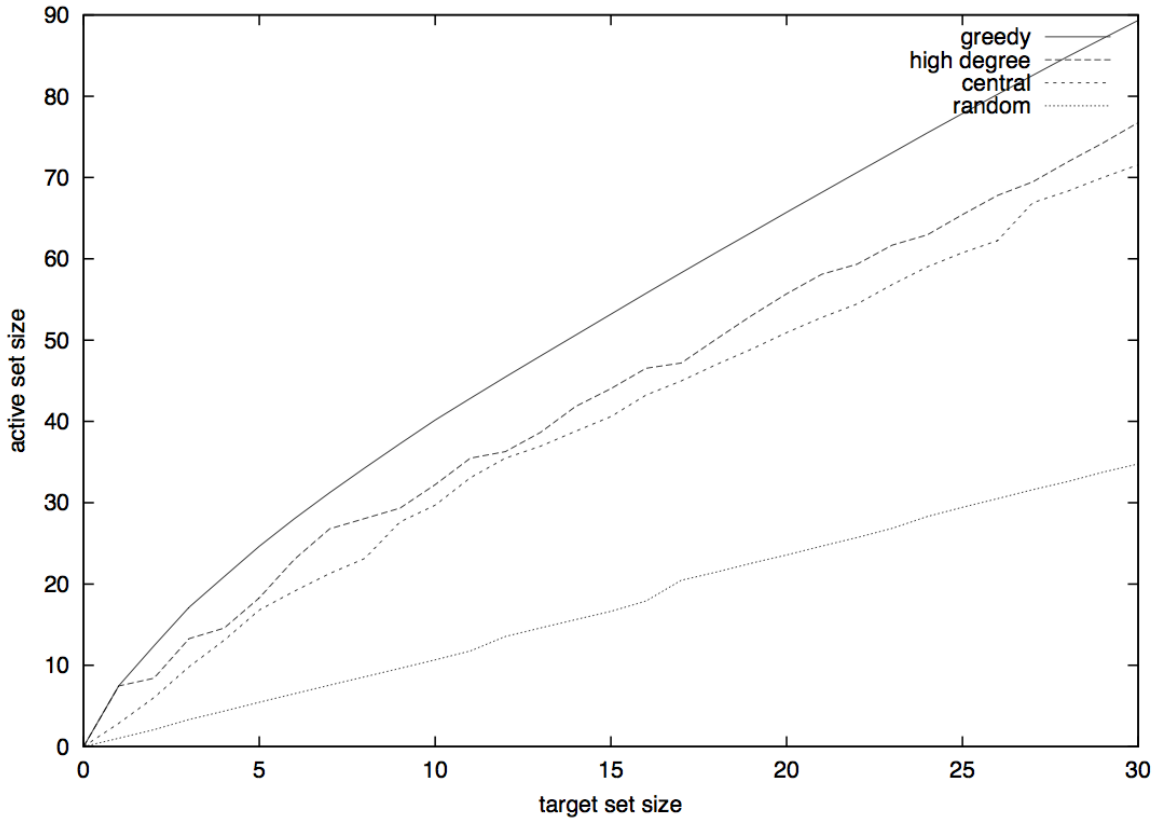
MIIA: `http://hanj.cs.illinois.edu/pdf/dmkd12_cwang.pdf/`

Tutorial: cf. Andreas Krause `http://submodularity.org/`

Course: Jeff Billmes at UW

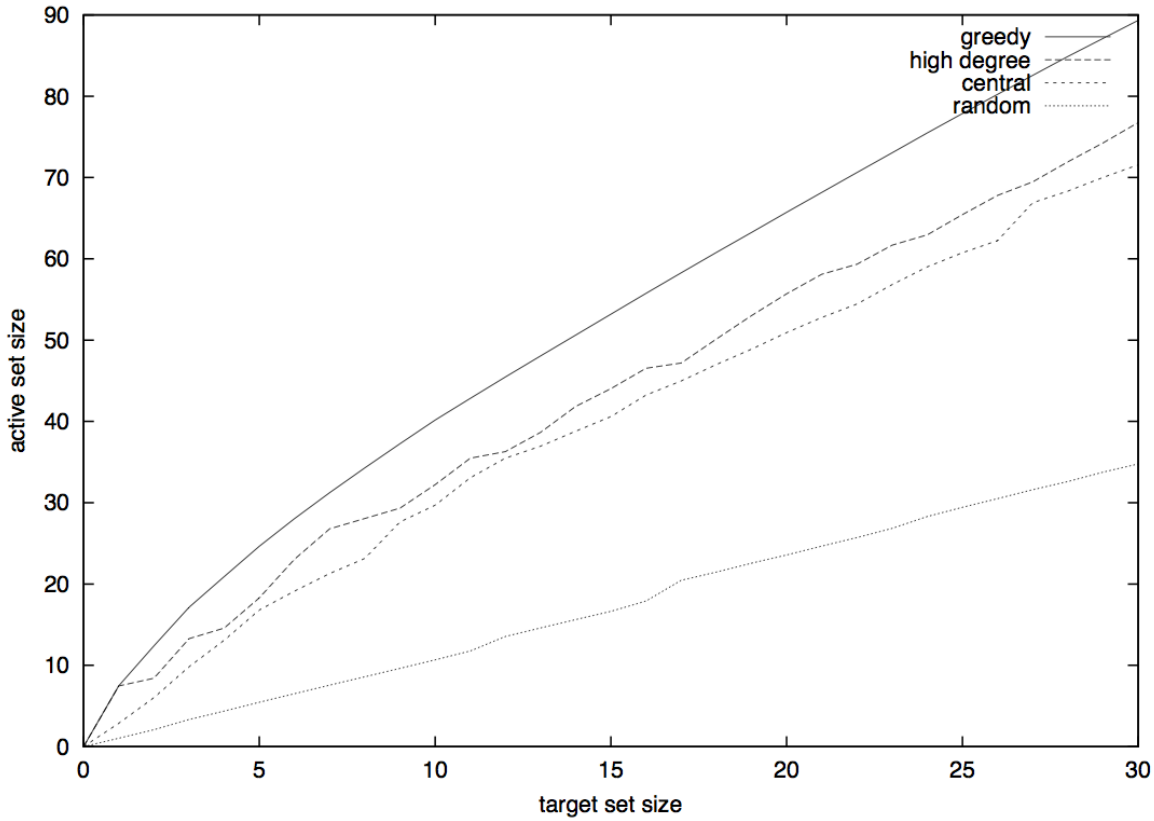
Success story #1 Product placement - comparison

influence on the ArXiv/Physics co-authorship graph



Success story #1 Product placement - comparison

influence on the ArXiv/Physics co-authorship graph



greedy approximation does better than the centrality measures

Natural graphs from utility and technology networks

- power grids, roads, Internet, sensor networks

Berkeley's Floating Sensor Network

Natural graphs from utility and technology networks

- power grids, roads, Internet, sensor networks
- structure is either *hand designed* or not

Berkeley's Floating Sensor Network

Natural graphs from utility and technology networks

- power grids, roads, Internet, sensor networks
- structure is either *hand designed* or not
- typical ML tasks

Berkeley's Floating Sensor Network

Natural graphs from utility and technology networks

- power grids, roads, Internet, sensor networks
- structure is either *hand designed* or not
- typical ML tasks
  - best routing under unknown or variable costs

Berkeley's Floating Sensor Network

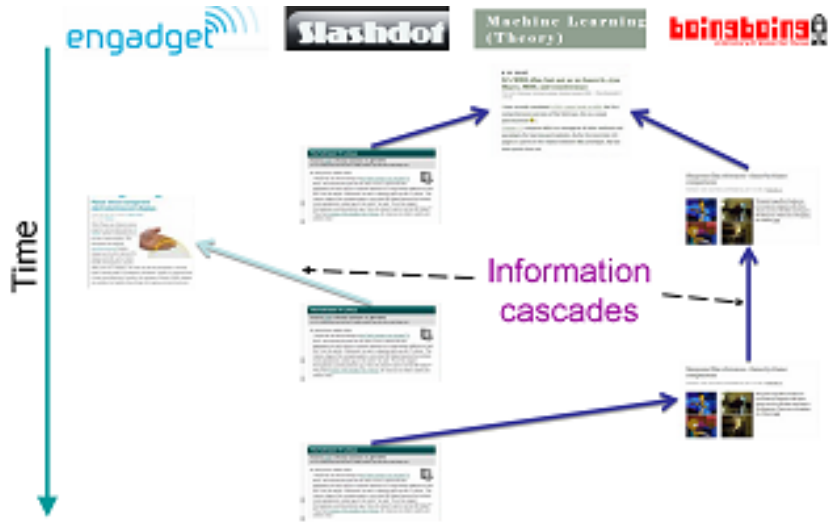
Natural graphs from utility and technology networks

- power grids, roads, Internet, sensor networks
- structure is either *hand designed* or not
- typical ML tasks
  - best routing under unknown or variable costs
  - identify the node of interest

Berkeley's Floating Sensor Network

Natural graphs from information networks

- web
- blogs
- wikipedia
- typical ML tasks
  - find influential sources
  - search (`PageRank`)



Blog cascades (ETH) - *submodularity*

Success story #2 Google PageRank

Objective: **Rank** all web pages (nodes on the graph) by how **many** other pages link to them and how **important** they are.

Success story #2 Google PageRank

Objective: **Rank** all web pages (nodes on the graph) by how **many** other pages link to them and how **important** they are.

basic PageRank is independent of query and the page content

Success story #2 Google PageRank

Objective: **Rank** all web pages (nodes on the graph) by how **many** other pages link to them and how **important** they are.

basic PageRank is independent of query and the page content

Internet

Success story #2 Google PageRank

Objective: **Rank** all web pages (nodes on the graph) by how **many** other pages link to them and how **important** they are.

basic PageRank is independent of query and the page content

Internet $\rightarrow$ graph

Success story #2 Google PageRank

Objective: **Rank** all web pages (nodes on the graph) by how **many** other pages link to them and how **important** they are.

basic PageRank is independent of query and the page content

Internet $\rightarrow$ graph $\rightarrow$ matrix

Success story #2 Google PageRank

Objective: **Rank** all web pages (nodes on the graph) by how **many** other pages link to them and how **important** they are.

basic PageRank is independent of query and the page content

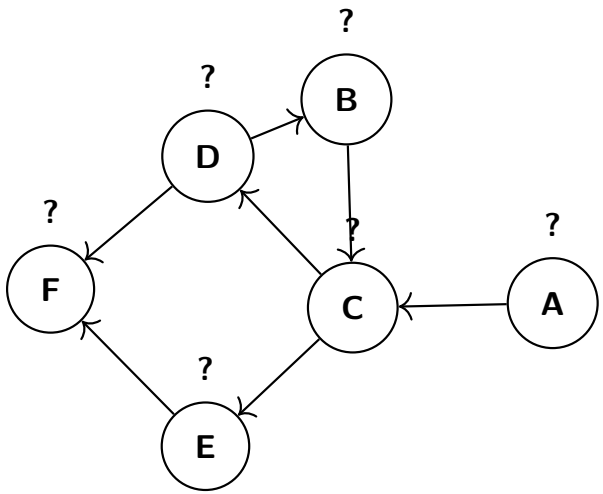
Internet $\rightarrow$ graph $\rightarrow$ matrix $\rightarrow$ stochastic matrix $\mathbf{M}$ $\left(\sum_j \mathbf{M}_{ij} = 1\right)$

Success story #2 Google PageRank

Objective: **Rank** all web pages (nodes on the graph) by how **many** other pages link to them and how **important** they are.

basic PageRank is independent of query and the page content

Internet $\rightarrow$ graph $\rightarrow$ matrix $\rightarrow$ stochastic matrix $\mathbf{M}$ $\left(\sum_j \mathbf{M}_{ij} = 1\right)$



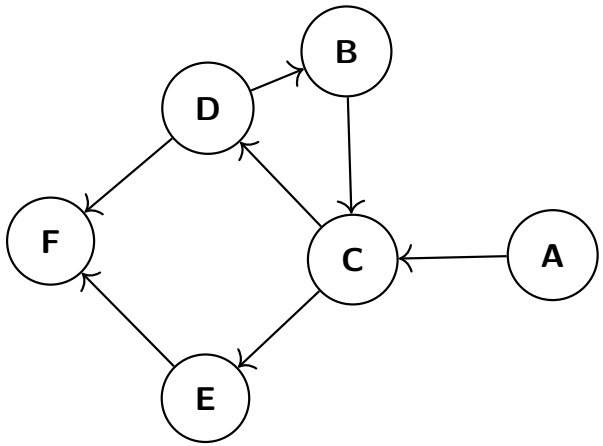
Success story #2 Goos eRank

Objective: **Rank** ph) by how **many** other pages li e.

basic PageRanl ge content

Internet → graph → trix $\mathbf{M}$ $\left(\sum_j \mathbf{M}_{ij} = 1\right)$

Random Surfer Process



Success story #2 Goos eRank

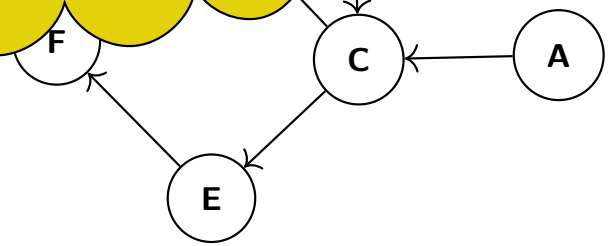
Objective: **Rank** ph) by how **many** other pages li e.

basic PageRanl ge content

Internet → graph → trix $\mathbf{M}$ $\left(\sum_j \mathbf{M}_{ij} = 1\right)$

Random Surfer Process

What is wrong with it?



Success story #2 Goos eRank

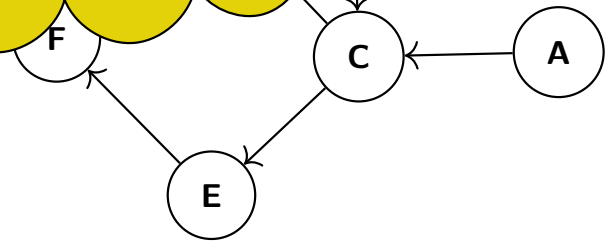
Objective: **Rank** ph) by how **many** other pages li e.

basic PageRanl ge content

Internet → graph → trix $\mathbf{M}$ $\left(\sum_j \mathbf{M}_{ij} = 1\right)$

Random Surfer Process

What is wrong with it?



dangling pages act like sinks

Success story #2 Google PageRank

`http://infolab.stanford.edu/~backrub/google.html`:

PageRank can be thought of as a model of user behavior. We assume there is a "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page.

Success story #2 Google PageRank

`http://infolab.stanford.edu/~backrub/google.html`:

PageRank can be thought of as a model of user behavior. We assume there is a "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page.

- page is **important** if **important** pages link **to** it

Success story #2 Google PageRank

`http://infolab.stanford.edu/~backrub/google.html`:

PageRank can be thought of as a model of user behavior. We assume there is a "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page.

- page is **important** if **important** pages link **to** it
  - circular definition

Success story #2 Google PageRank

`http://infolab.stanford.edu/~backrub/google.html`:

PageRank can be thought of as a model of user behavior. We assume there is a "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page.

- page is **important** if **important** pages link **to** it
  - circular definition
- importance of a page is distributed **evenly**

Success story #2 Google PageRank

`http://infolab.stanford.edu/~backrub/google.html`:

PageRank can be thought of as a model of user behavior. We assume there is a "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page.

- page is **important** if **important** pages link **to** it
  - circular definition
- importance of a page is distributed **evenly**
- probability of being bored is 15%

Success story #2 Google PageRank

Google matrix: $\mathbf{G} = (1-p)\mathbf{M} + p \cdot \frac{1}{N}\mathbb{1}_{N\times N}$, where $p = 0.15$

Success story #2 Google PageRank

Google matrix: $\mathbf{G} = (1-p)\mathbf{M} + p \cdot \frac{1}{N}\mathbb{1}_{N\times N}$, where $p = 0.15$

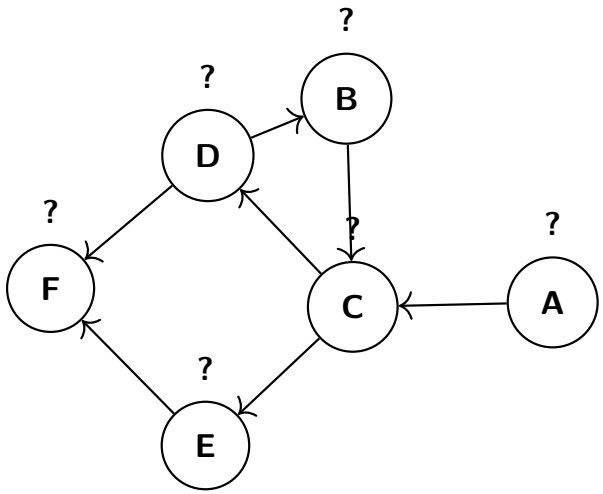
G is **stochastic** why? What is **Ga** for any **a**? We look for $\mathbf{Gv} = 1 \times \mathbf{v}$, steady-state vector, a right eigenvector with eigenvalue 1. why?

Success story #2 Google PageRank

Google matrix: $\mathbf{G} = (1-p)\mathbf{M} + p \cdot \frac{1}{N}\mathbb{1}_{N\times N}$, where $p = 0.15$

G is **stochastic** why? What is **Ga** for any **a**? We look for $\mathbf{Gv} = 1 \times \mathbf{v}$, steady-state vector, a right eigenvector with eigenvalue 1. why?

Perron's theorem: Such v exists and it is **unique** if the entries of **G** are positive.

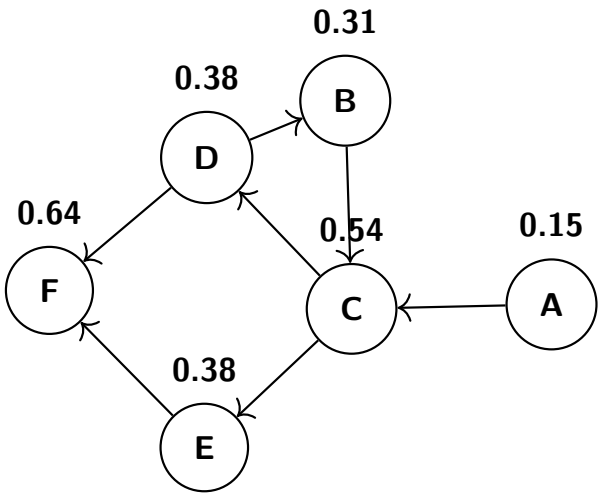


Success story #2 Google PageRank

Google matrix: $\mathbf{G} = (1-p)\mathbf{M} + p \cdot \frac{1}{N}\mathbb{1}_{N\times N}$, where $p = 0.15$

G is **stochastic** why? What is **Ga** for any **a**? We look for $\mathbf{Gv} = 1 \times \mathbf{v}$, steady-state vector, a right eigenvector with eigenvalue 1. why?

Perron's theorem: Such v exists and it is **unique** if the entries of **G** are positive.

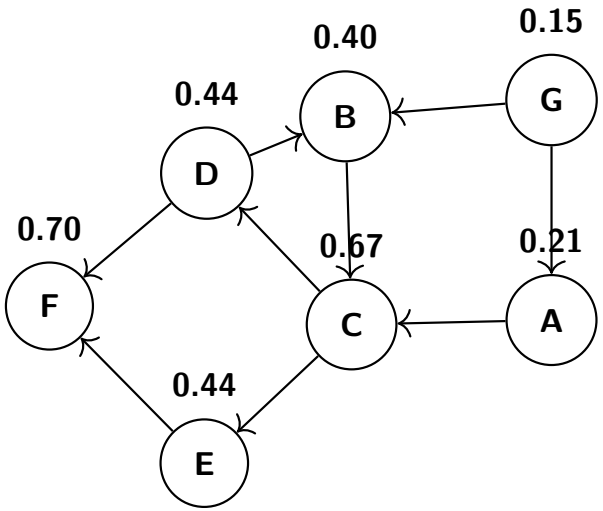


Success story #2 Google PageRank

Google matrix: $\mathbf{G} = (1-p)\mathbf{M} + p \cdot \frac{1}{N}\mathbb{1}_{N\times N}$, where $p = 0.15$

G is **stochastic** why? What is **Ga** for any **a**? We look for $\mathbf{Gv} = 1 \times \mathbf{v}$, steady-state vector, a right eigenvector with eigenvalue 1. why?

Perron's theorem: Such v exists and it is **unique** if the entries of **G** are positive.



Success story #2 Google PageRank

History: [Desikan, 2006]

- The anatomy of a large-scale hypertextual web search engine [Brin & Page 1998]

Success story #2 Google PageRank

History: [Desikan, 2006]

- The anatomy of a large-scale hypertextual web search engine [Brin & Page 1998]
- US patent for PageRank granted in 2001

Success story #2 Google PageRank

History: [Desikan, 2006]

- The anatomy of a large-scale hypertextual web search engine [Brin & Page 1998]
- US patent for PageRank granted in 2001
- Google indexes 10's of billions of web pages (1 billion $= 10^9$)

Success story #2 Google PageRank

History: [Desikan, 2006]

- The anatomy of a large-scale hypertextual web search engine [Brin & Page 1998]
- US patent for PageRank granted in 2001
- Google indexes 10's of billions of web pages (1 billion $= 10^9$)
- Google serves $\geq$ 200 million queries per day

Success story #2 Google PageRank

History: [Desikan, 2006]

- The anatomy of a large-scale hypertextual web search engine [Brin & Page 1998]
- US patent for PageRank granted in 2001
- Google indexes 10's of billions of web pages (1 billion $= 10^9$)
- Google serves $\geq$ 200 million queries per day
- Each query processed by $\geq$ 1000 machines

Success story #2 Google PageRank

History: [Desikan, 2006]

- The anatomy of a large-scale hypertextual web search engine [Brin & Page 1998]
- US patent for PageRank granted in 2001
- Google indexes 10's of billions of web pages (1 billion $= 10^9$)
- Google serves $\geq$ 200 million queries per day
- Each query processed by $\geq$ 1000 machines
- All search engines combined process more than 500 million queries per day

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse**

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse** (average outdegree: 7)

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse** (average outdegree: 7)
- power method

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse** (average outdegree: 7)
- power method

$$\mathbf{v}_0 = (1_A \quad 0_B \quad 0_C \quad 0_D \quad 0_E \quad 0_F)^\top$$

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse** (average outdegree: 7)
- power method

$$\mathbf{v}_0 = (1_A \quad 0_B \quad 0_C \quad 0_D \quad 0_E \quad 0_F)^\mathsf{T}$$

$$\mathbf{v}_1 = \mathbf{G}\mathbf{v}_0$$

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse** (average outdegree: 7)
- power method

$$\mathbf{v}_0 = (1_A \quad 0_B \quad 0_C \quad 0_D \quad 0_E \quad 0_F)^\mathsf{T}$$

$$\mathbf{v}_1 = \mathbf{G}\mathbf{v}_0$$

$$\mathbf{v}_{t+1} = \mathbf{G}\mathbf{v}_t$$

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse** (average outdegree: 7)
- power method

$$\mathbf{v}_0 = (1_A \quad 0_B \quad 0_C \quad 0_D \quad 0_E \quad 0_F)^\mathsf{T}$$

$$\mathbf{v}_1 = \mathbf{G}\mathbf{v}_0$$

$$\mathbf{v}_{t+1} = \mathbf{G}\mathbf{v}_t$$

$$\mathbf{v}_{t+1} = \mathbf{v}_t \implies \mathbf{G}\mathbf{v}_t = \mathbf{v}_t$$

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse** (average outdegree: 7)
- power method

$$\mathbf{v}_0 = (1_A \quad 0_B \quad 0_C \quad 0_D \quad 0_E \quad 0_F)^\mathsf{T}$$

$$\mathbf{v}_1 = \mathbf{G}\mathbf{v}_0$$

$$\mathbf{v}_{t+1} = \mathbf{G}\mathbf{v}_t$$

$\mathbf{v}_{t+1} = \mathbf{v}_t \implies \mathbf{G}\mathbf{v}_t = \mathbf{v}_t$ and we found the steady vector

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse** (average outdegree: 7)
- power method

$$\mathbf{v}_0 = (1_A \quad 0_B \quad 0_C \quad 0_D \quad 0_E \quad 0_F)^\mathsf{T}$$

$$\mathbf{v}_1 = \mathbf{G}\mathbf{v}_0$$

$$\mathbf{v}_{t+1} = \mathbf{G}\mathbf{v}_t$$

$\mathbf{v}_{t+1} = \mathbf{v}_t \implies \mathbf{G}\mathbf{v}_t = \mathbf{v}_t$ and we found the steady vector

But wait, **M** is sparse, but **G** is dense! What to do?

Success story #2 Google PageRank

Problem: Find an eigenvector of a stochastic matrix.

- $n = 10^9$!!!
- luckily: **sparse** (average outdegree: 7)
- power method

$$\mathbf{v}_0 = (1_A \quad 0_B \quad 0_C \quad 0_D \quad 0_E \quad 0_F)^\mathsf{T}$$

$$\mathbf{v}_1 = \mathbf{G}\mathbf{v}_0$$

$$\mathbf{v}_{t+1} = \mathbf{G}\mathbf{v}_t$$

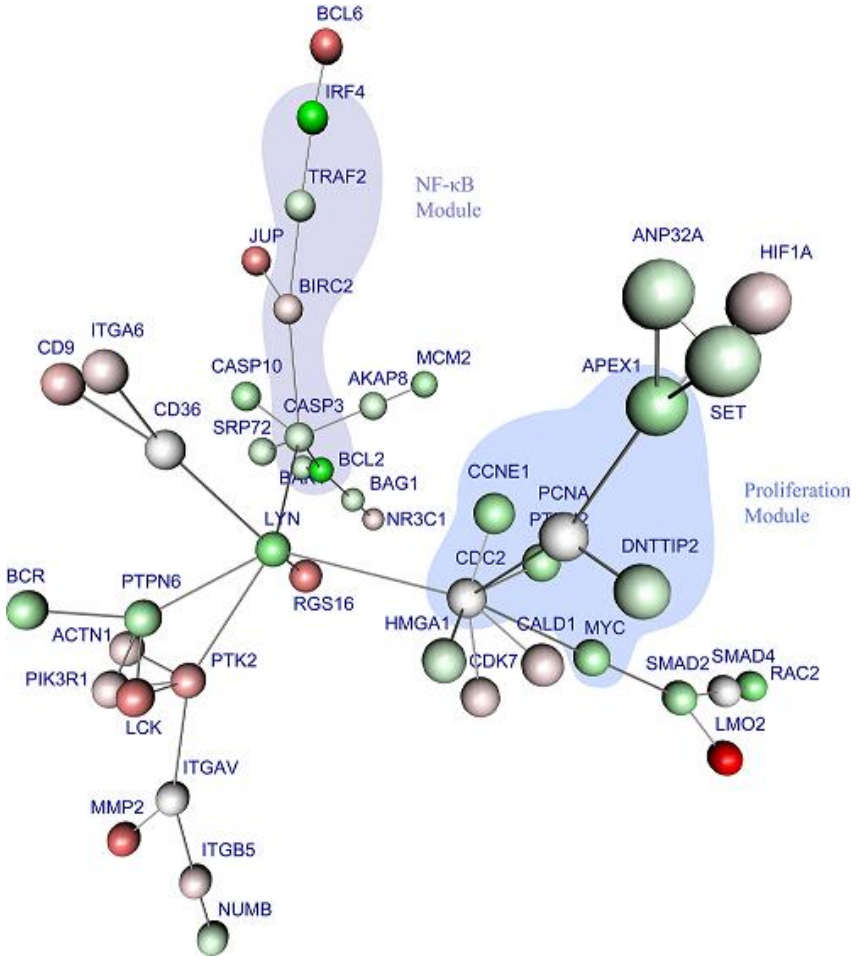
$\mathbf{v}_{t+1} = \mathbf{v}_t \implies \mathbf{G}\mathbf{v}_t = \mathbf{v}_t$ and we found the steady vector

But wait, **M** is sparse, but **G** is dense! What to do?

we store only **M** but do computations as with **G**

Natural graphs from biological networks

- protein-protein interactions
- gene regulatory networks
- typical ML tasks
  - discover unexplored interactions
  - learn or reconstruct the structure



Diffuse large B-cell lymphomas - Dittrich et al. (2008)

Sources of Real Networks

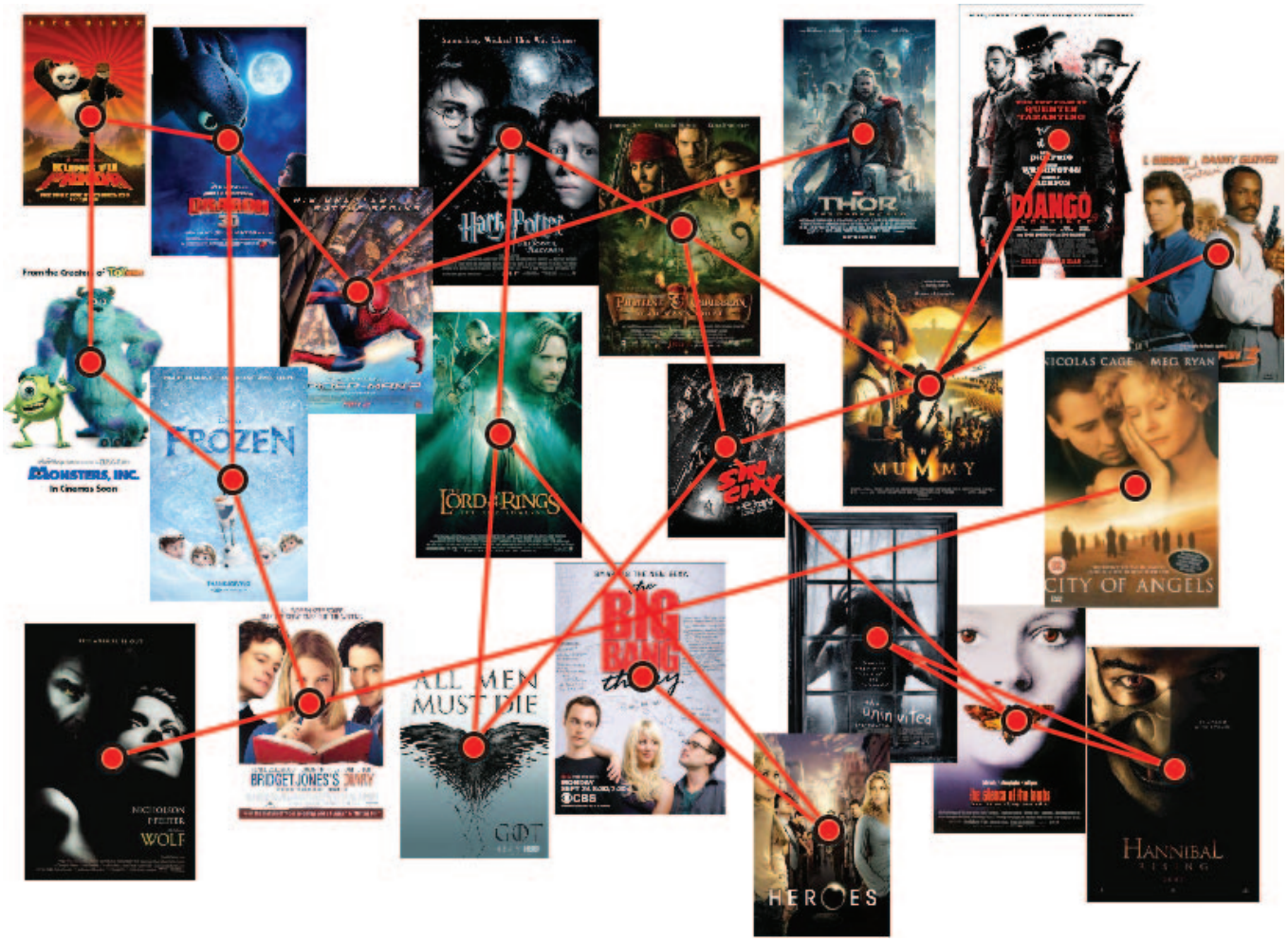
- https://ogb.stanford.edu/
- http://snap.stanford.edu/data/
- http://www-personal.umich.edu/~mejn/netdata/
- http://proj.ise.bgu.ac.il/sns/datasets.html
- http://www.cise.ufl.edu/research/sparse/matrices/
- http://vlado.fmf.uni-lj.si/pub/networks/data/default.htm

Constructed graphs from similarity networks

graph is not naturally given

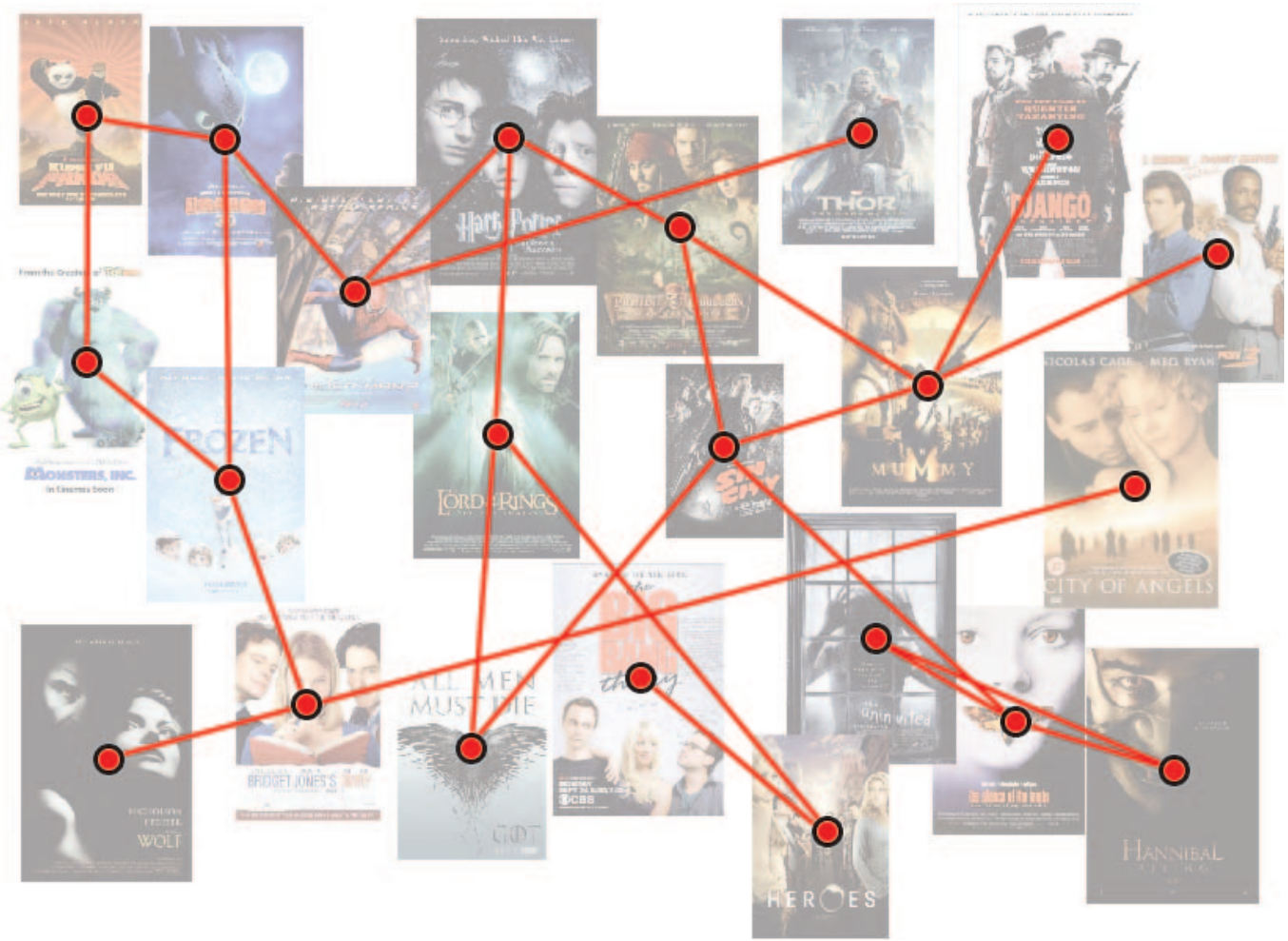
Constructed graphs from similarity networks

but we can construct it



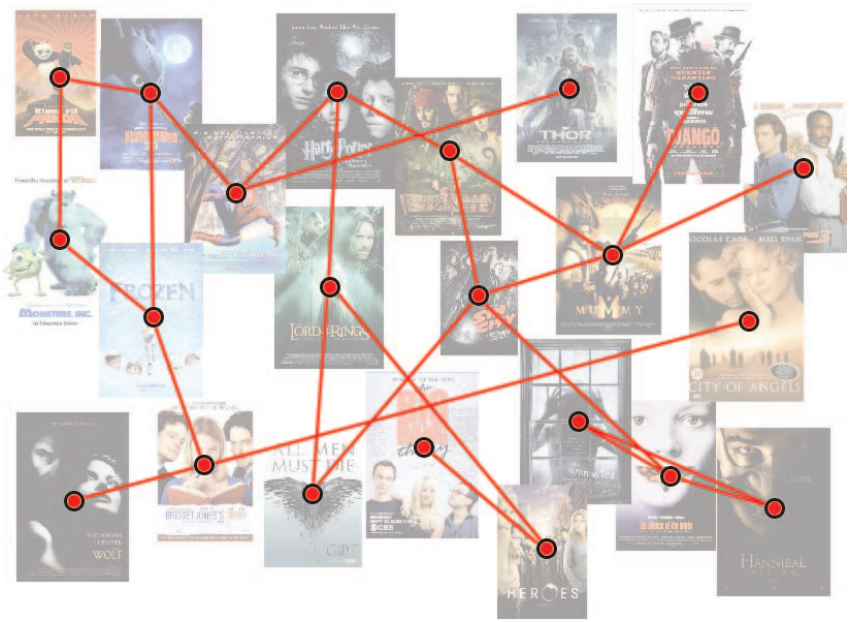
Constructed graphs from similarity networks

and use it as an abstraction



Constructed graphs from similarity networks

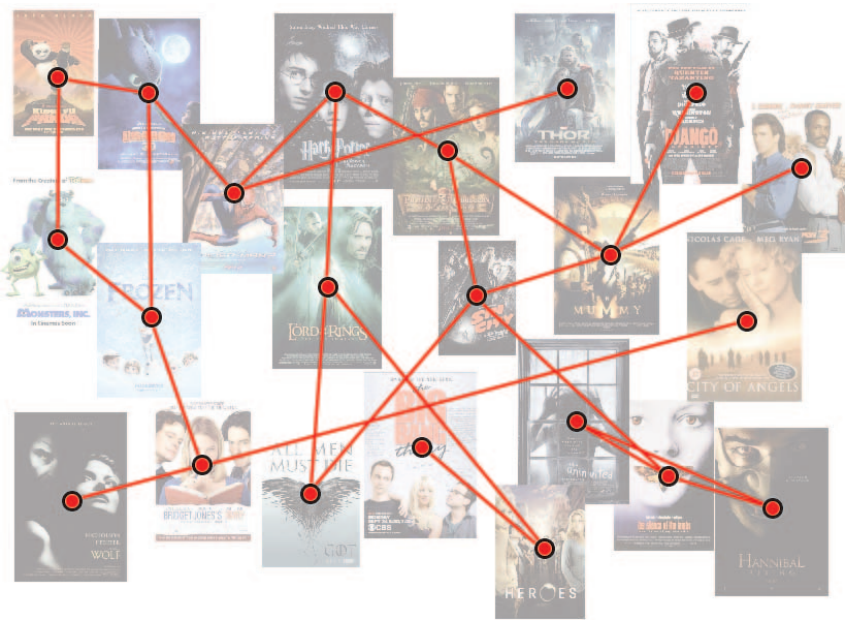
- vision
- audio
- text



movie similarity

Constructed graphs from similarity networks

- vision
- audio
- text
- typical ML tasks
  - **semi-supervised learning**
  - **spectral clustering**
  - **manifold learning**

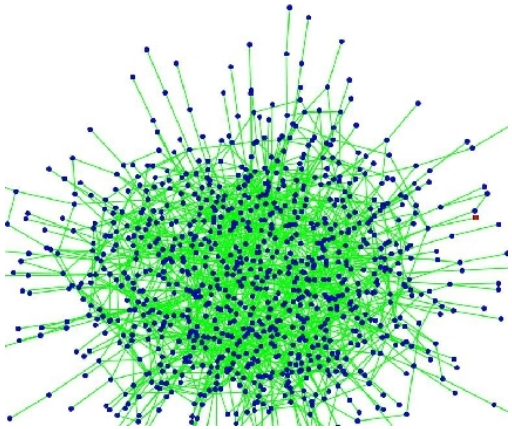


movie similarity

The in-between option: random graph models

Erdős-Rényi

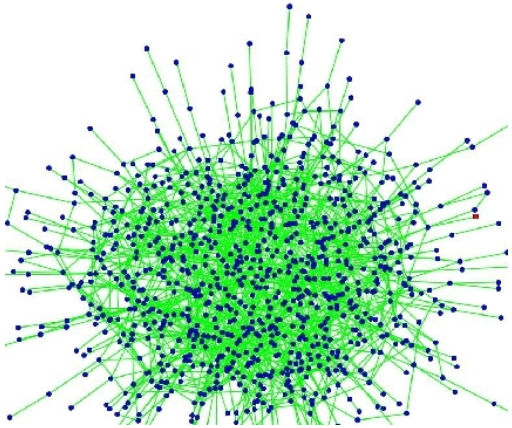
independent edges



The in-between option: random graph models

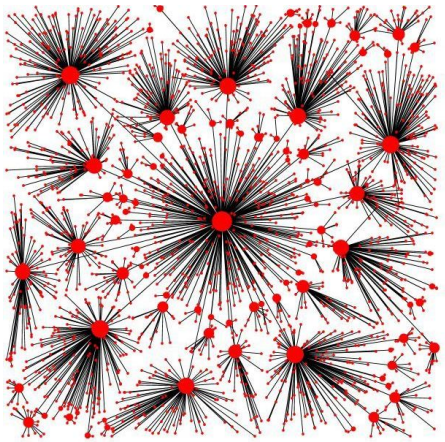
Erdős-Rényi

independent edges



Barabási-Albert

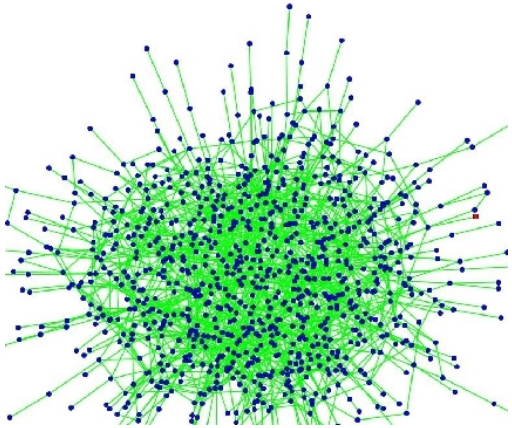
preferential attachment



The in-between option: random graph models

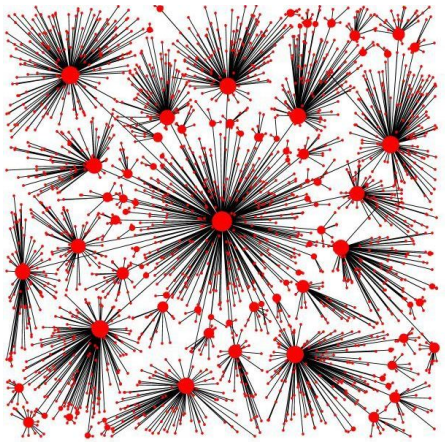
Erdős-Rényi

independent edges



Barabási-Albert

preferential attachment

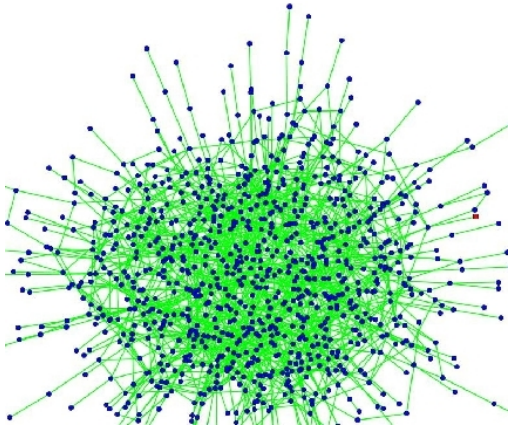


Stochastic Blocks

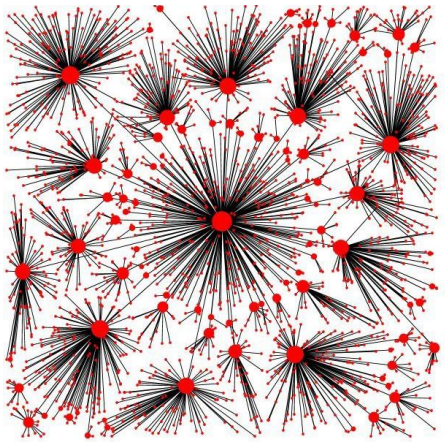
modeling communities

The in-between option: random graph models

Erdős-Rényi
independent edges

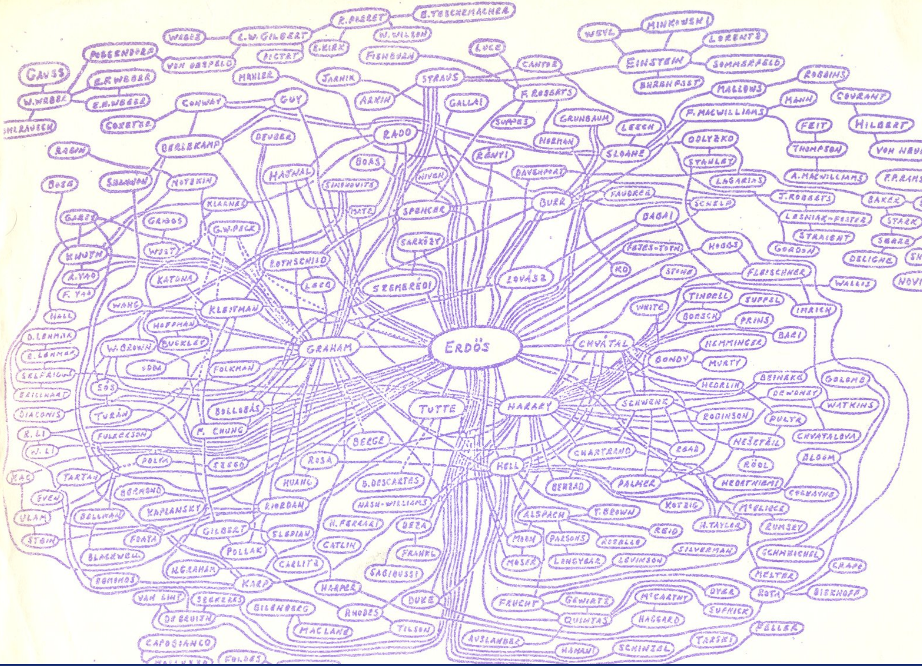


Barabási-Albert
preferential attachment



Stochastic Blocks
modeling communities

Watts-Strogatz, Chung-Lu, Fiedler, ...



Ron Graham (alias Tom Odda).

Erdős number project

- http://www.oakland.edu/enp/ try it!

Erdős number project

- `http://www.oakland.edu/enp/` **try it!**
- an example of a real-world graph

Erdős number project

- `http://www.oakland.edu/enp/` **try it!**
- an example of a real-world graph
- 401 000 authors, 676 000 edges ($\ll 401000^2 \rightarrow$ sparse)

Erdős number project

- `http://www.oakland.edu/enp/` **try it!**
- an example of a real-world graph
- 401 000 authors, 676 000 edges ($\ll 401000^2 \rightarrow$ sparse)
- average degree 3.36

Erdős number project

- `http://www.oakland.edu/enp/` **try it!**
- an example of a real-world graph
- 401 000 authors, 676 000 edges ($\ll 401000^2 \rightarrow$ sparse)
- average degree 3.36
- average distance for the largest component: 7.64

Erdős number project

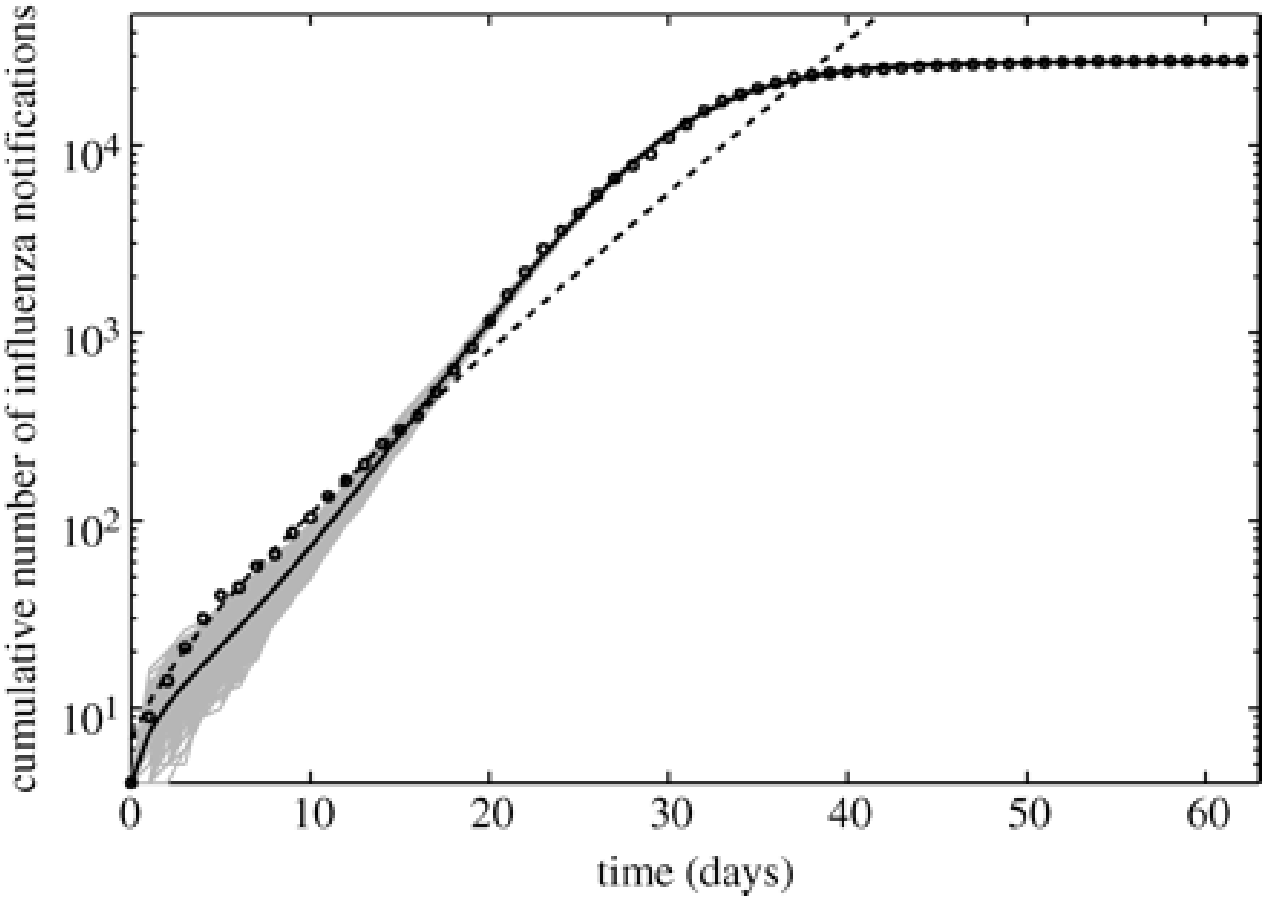
- `http://www.oakland.edu/enp/` **try it!**
- an example of a real-world graph
- 401 000 authors, 676 000 edges ($\ll 401000^2 \rightarrow$ sparse)
- average degree 3.36
- average distance for the largest component: 7.64
- 6 degrees of separation [Travers & Milgram, 1967]

Erdős number project

- `http://www.oakland.edu/enp/` **try it!**
- an example of a real-world graph
- 401 000 authors, 676 000 edges ($\ll 401000^2 \rightarrow$ sparse)
- average degree 3.36
- average distance for the largest component: 7.64
- 6 degrees of separation [Travers & Milgram, 1967]
- heavy tail

Spanish flu in San Francisco 1918–1919

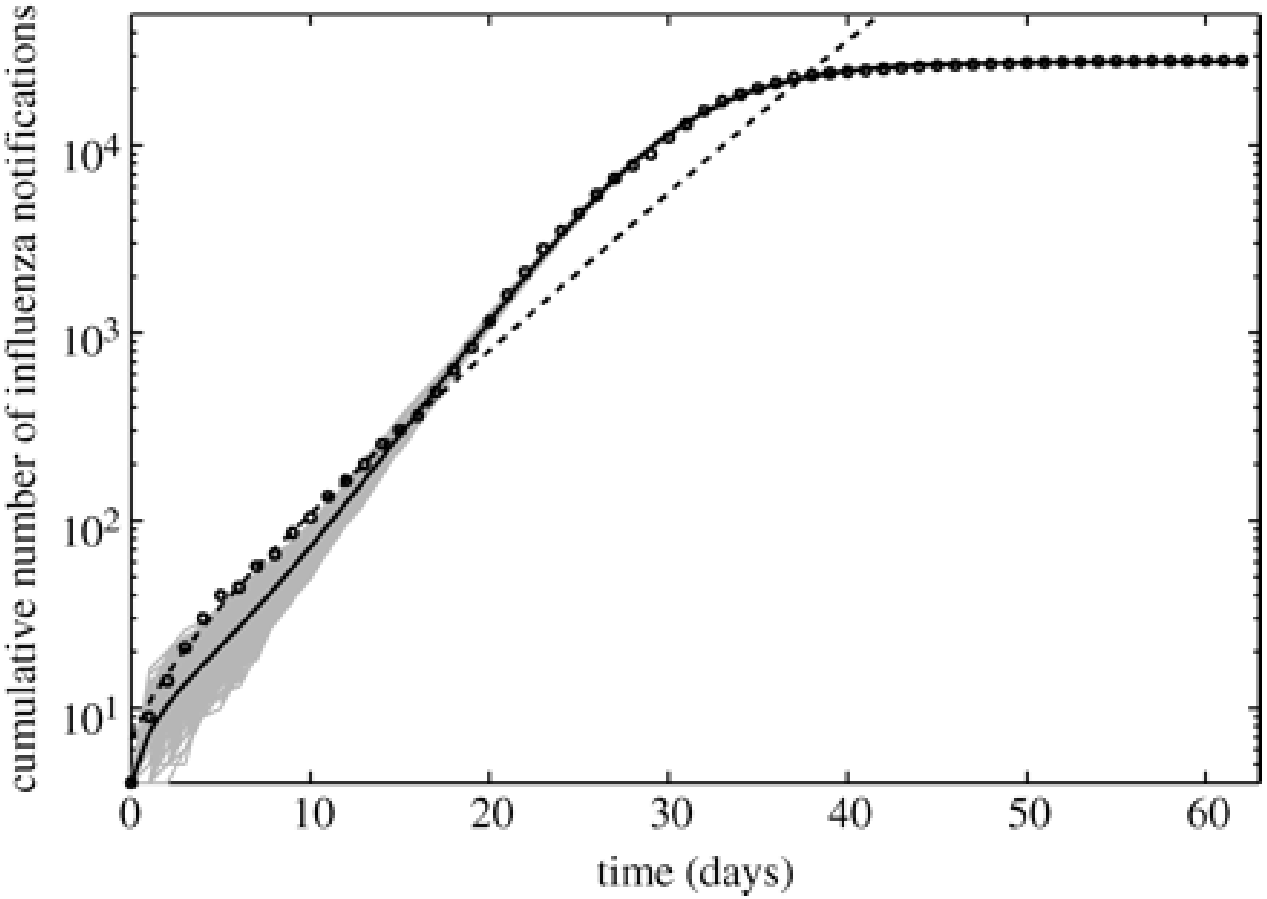
Small-world phenomenon and diseases



http://rsif.royalsocietypublishing.org/content/4/12/155

Spanish flu in San Francisco 1918–1919

Small-world phenomenon and diseases

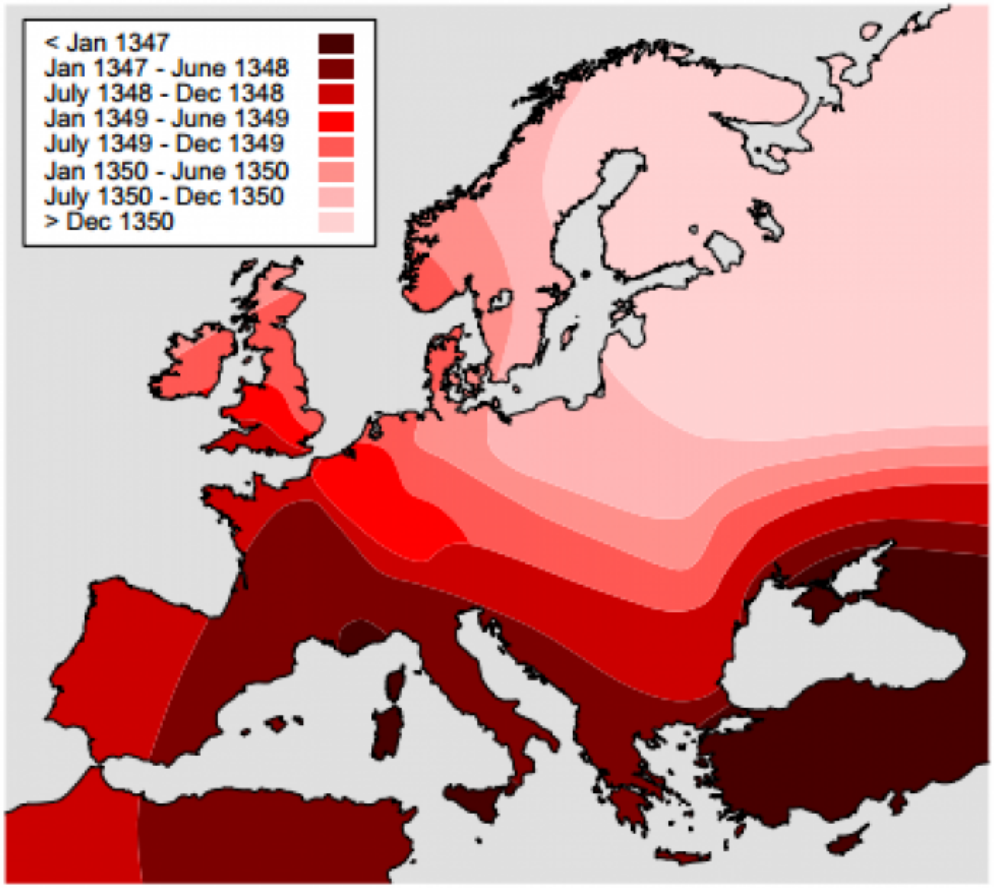


http://rsif.royalsocietypublishing.org/content/4/12/155

Small world: Obvious?

Black death!

Black death: spread



source: catholic.org

https://www.youtube.com/watch?v=EEK6c9Bh5CQ

What will you learn in the Graphs in ML course?

Concepts

What will you learn in the Graphs in ML course?

Concepts, tools

What will you learn in the Graphs in ML course?

Concepts, tools, and methods

What will you learn in the Graphs in ML course?

Concepts, tools, and methods to work with graphs in ML.

What will you learn in the Graphs in ML course?

Concepts, tools, and methods to work with graphs in ML.

Specific applications of graphs in ML.

What will you learn in the Graphs in ML course?

Concepts, tools, and methods to work with graphs in ML.

Specific applications of graphs in ML.

Theoretical toolbox to analyze graph-based algorithms.

What will you learn in the Graphs in ML course?

Concepts, tools, and methods to work with graphs in ML.

Specific applications of graphs in ML.

Theoretical toolbox to analyze graph-based algorithms.

How to tackle: *large graphs, online setting, graph construction* ...

What will you learn in the Graphs in ML course?

Concepts, tools, and methods to work with graphs in ML.

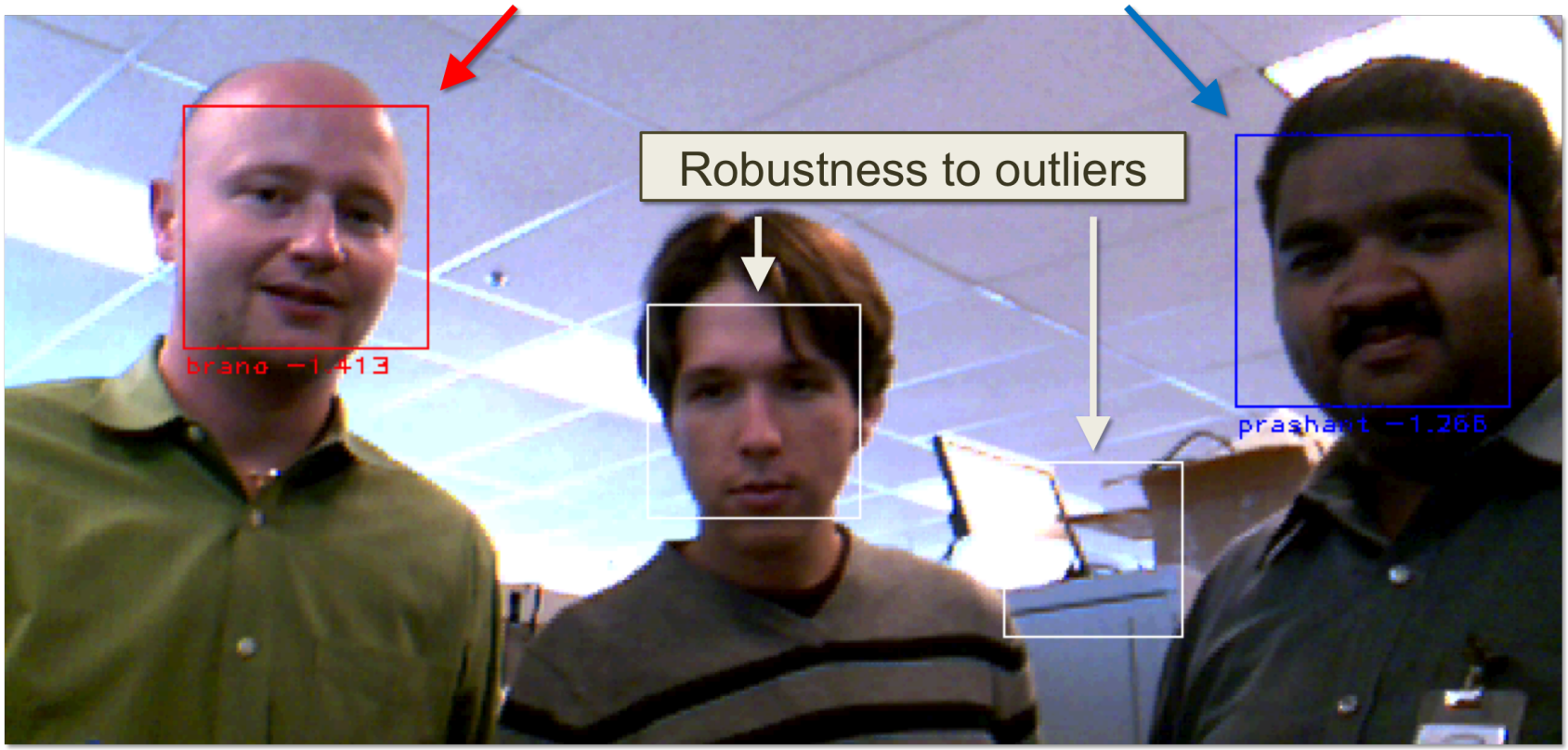
Specific applications of graphs in ML.

Theoretical toolbox to analyze graph-based algorithms.

How to tackle: *large graphs, online setting, graph construction* ...

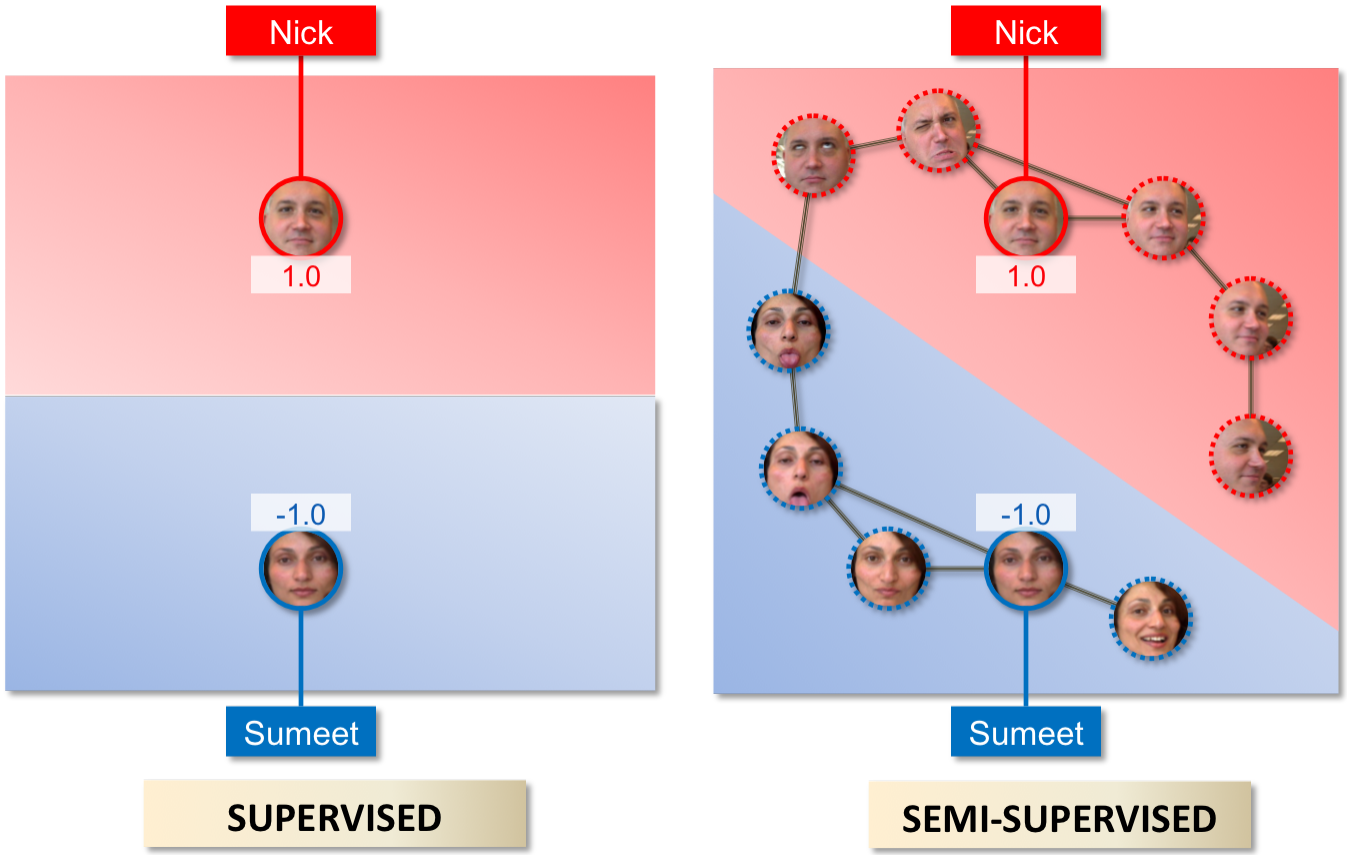
One example: Online Semi-Supervised Face Recognition

Online Semi-Supervised Face Recognition



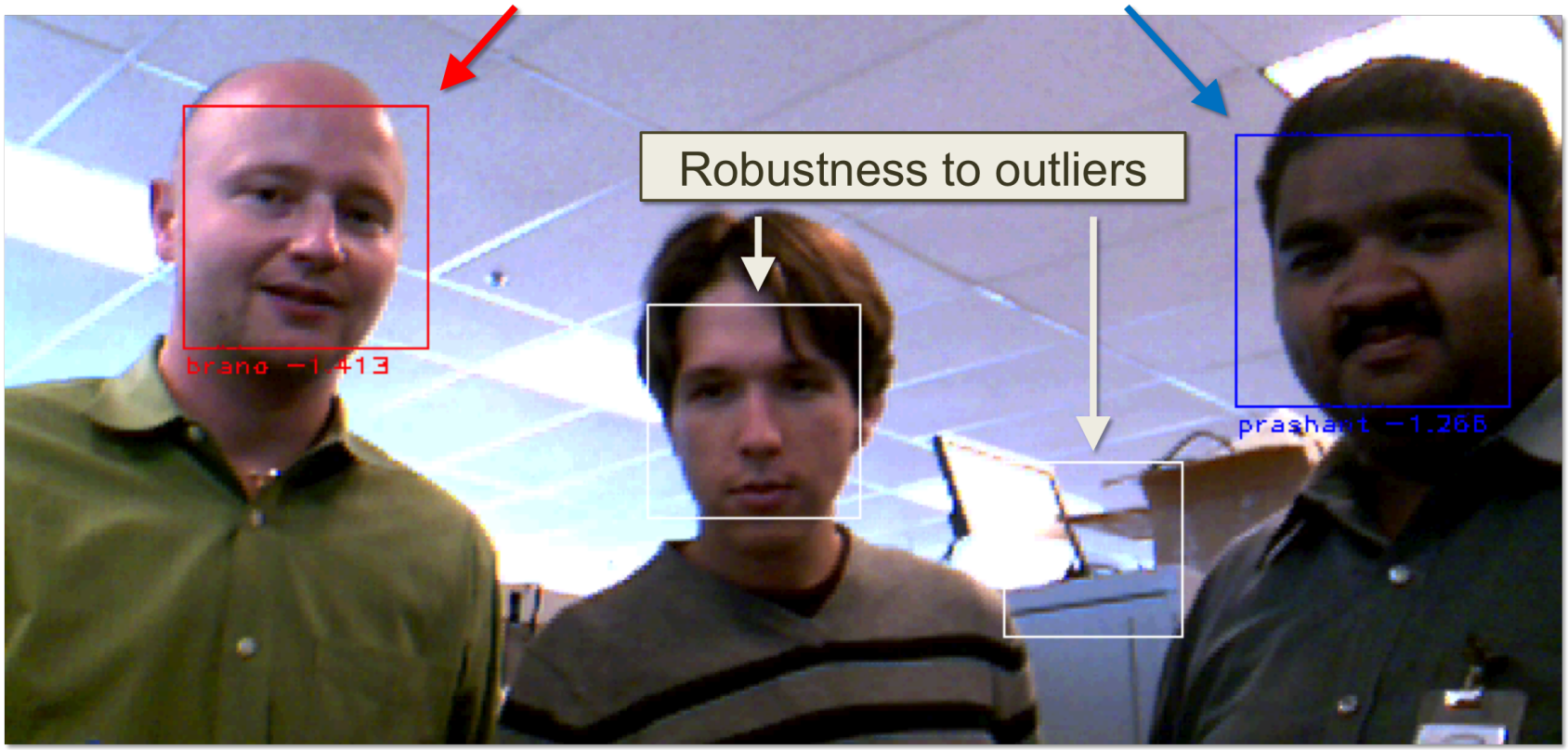
Online Semi-Supervised Face Recognition

graph-based semi-supervised learning



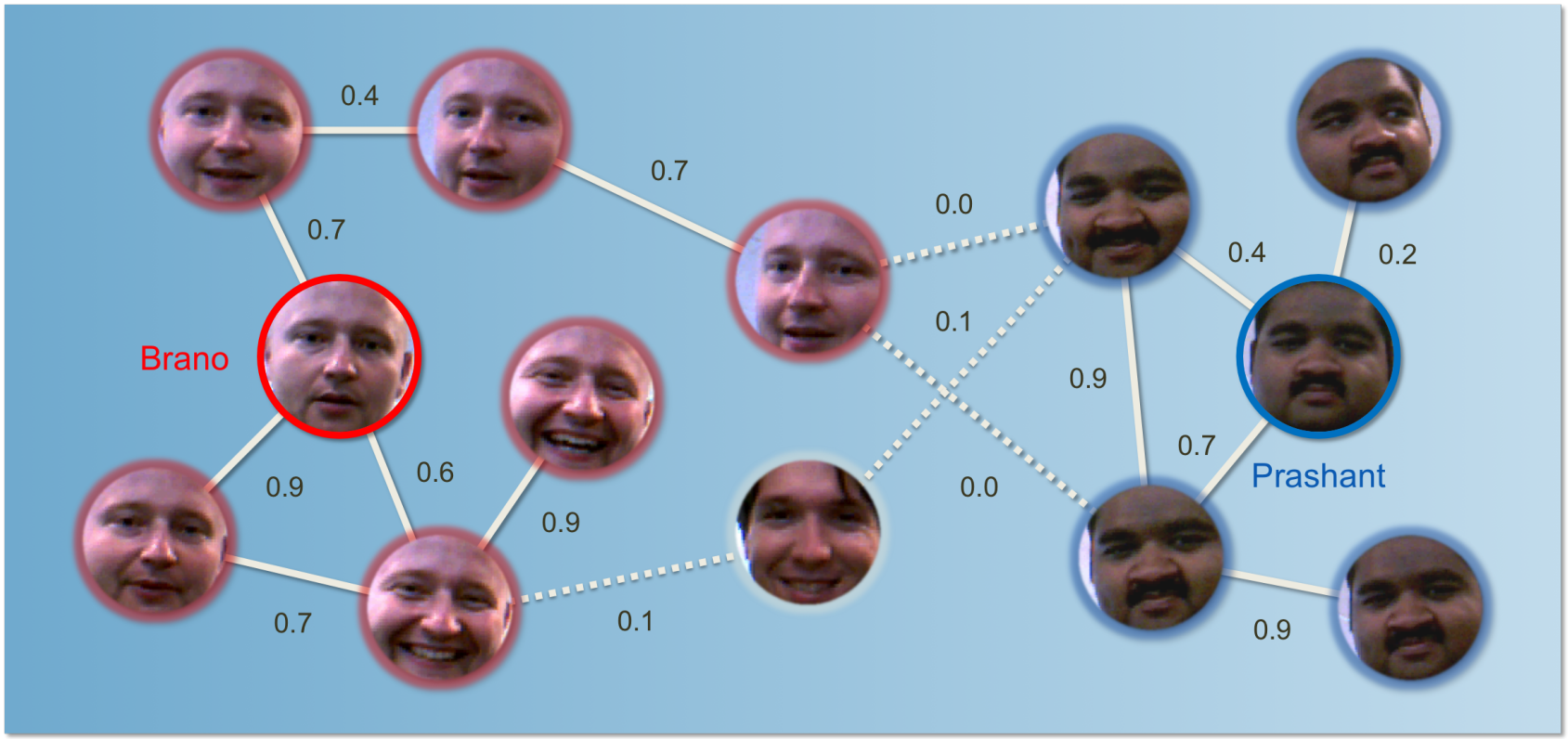
Online Semi-Supervised Face Recognition

graph is not given



Online Semi-Supervised Face Recognition

we will construct it!

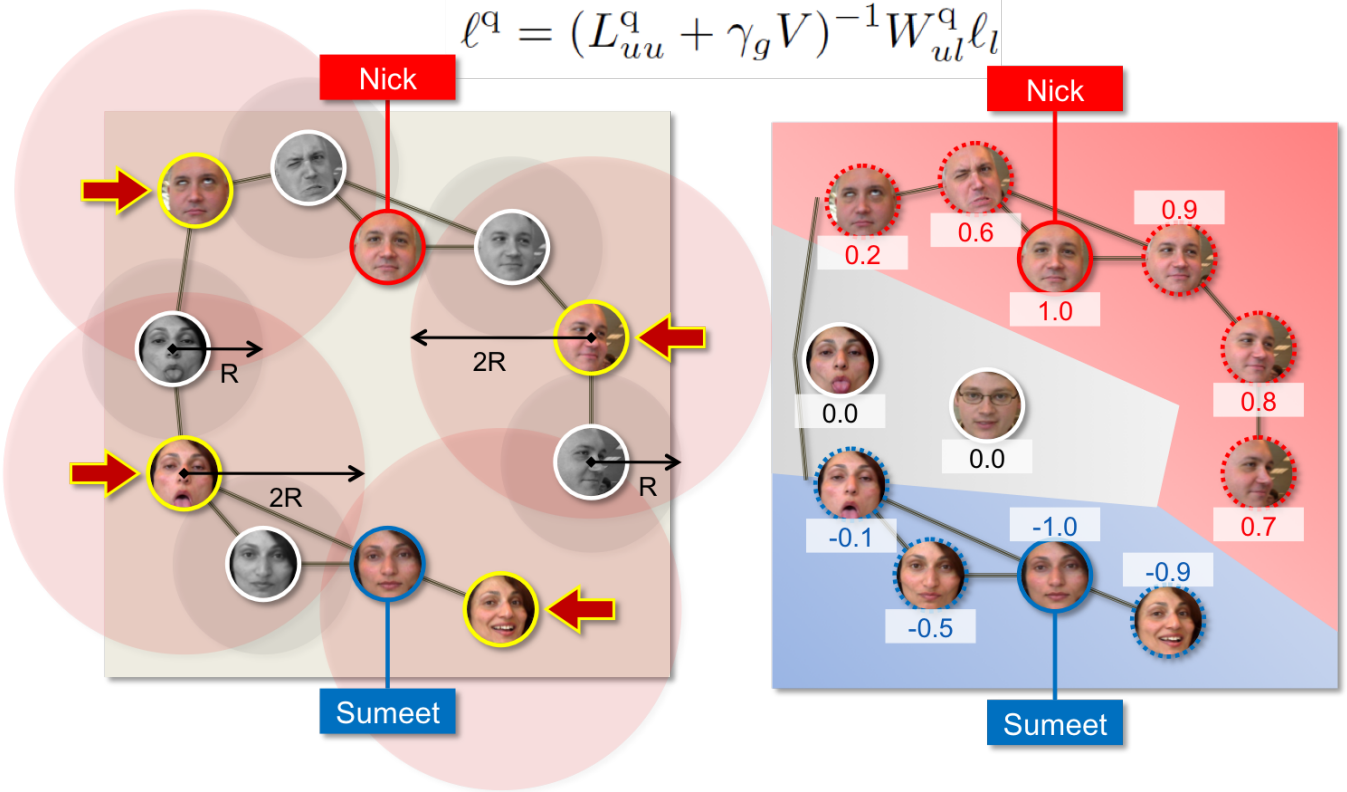


An example of a similarity graph over faces. The faces are vertices of the graph. The edges of the graph connect similar faces. Labeled faces are outlined by thick solid lines.

Online Semi-Supervised Face Recognition

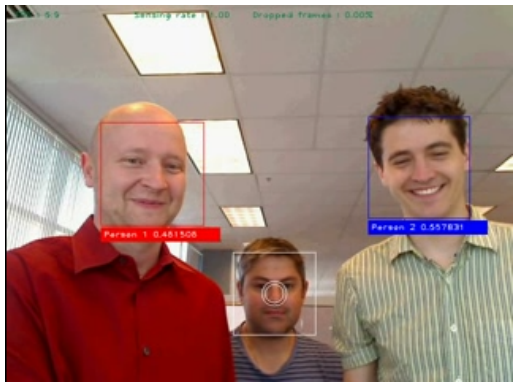
online learning - graph sparsification

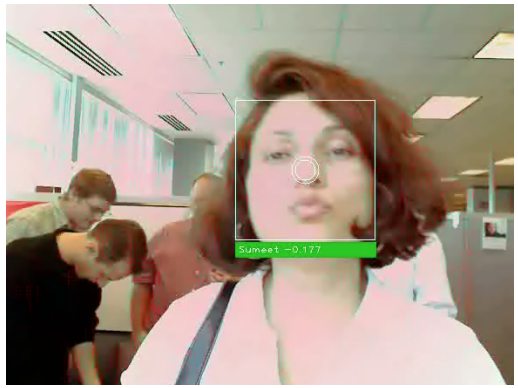
$$\ell^{\mathrm{q}} = (L^{\mathrm{q}}_{uu} + \gamma_g V)^{-1} W^{\mathrm{q}}_{ul} \ell_l$$



DEMO

second TD





see the demo: `http://researchers.lille.inria.fr/~valko/hp/serve.php?what=publications/kveton2009nipsdemo.officespace.mov`

OSS FaceReco: Analysis

$$\frac{1}{n}\sum_t (\ell_t^q[t]-y_t)^2 \le \frac{3}{n}\sum_t (\ell_t^*-y_t)^2 + \frac{3}{n}\sum_t (\ell_t^o[t]-\ell_t^*)^2 + \frac{3}{n}\sum_t (\ell_t^q[t]-\ell_t^o[t])^2$$

Error of our solution | Offline learning error | Online learning error | Quantization error

Claim: When the regularization parameter is set as $\gamma_g = \Omega(n_l^{3/2})$, the difference between the risks on labeled and all vertices decreases at the rate of $O(n_l^{-1/2})$ (with a high probability)

$$\frac{1}{n}\sum_t (\ell_t^*-y_t)^2 \le \frac{1}{n_l}\sum_{i\in\mathcal{L}} (\ell_i^*-y_i)^2 + \beta + \sqrt{\frac{2\ln(2/\delta)}{n_l}}\left(n_l\beta+4\right)$$

$$\beta \le \left[\frac{\sqrt{2}}{\gamma_g+1} + \sqrt{2n_l}\,\frac{1-\sqrt{c_u}}{\sqrt{c_u}}\,\frac{\lambda_M(L)+\gamma_g}{\gamma_g^2+1}\right]$$

OSS FaceReco: Analysis

$$\frac{1}{n}\sum_t (\ell_t^q[t]-y_t)^2 \le \frac{3}{n}\sum_t (\ell_t^*-y_t)^2 + \frac{3}{n}\sum_t (\ell_t^o[t]-\ell_t^*)^2 + \frac{3}{n}\sum_t (\ell_t^q[t]-\ell_t^o[t])^2$$

Error of our solution	Offline learning error	Online learning error	Quantization error

Claim: When the regularization parameter is set as $\gamma_g = \Omega(n^{1/4})$, the average error between the offline and online HFS predictions decreases at the rate of $O(n^{-1/2})$

$$\frac{1}{n}\sum_t (\ell_t^o[t]-\ell_t^*)^2 \le \frac{1}{n}\sum_t \left\|\ell^o[t]-\ell^*\right\|_2^2 \le \frac{4n_l}{(\gamma_g+1)^2}$$

$$\left\|\ell\right\|_2 \le \frac{\left\|y\right\|_2}{\lambda_m(C^{-1}K+I)} = \frac{\left\|y\right\|_2}{\lambda_m(K)\lambda_M^{-1}(C)+1} \le \frac{\sqrt{n_l}}{\gamma_g+1}$$

OSS FaceReco: Analysis

$$\frac{1}{n}\sum_t (\ell_t^{\mathrm{q}}[t]-y_t)^2 \le \frac{3}{n}\sum_t (\ell_t^{*}-y_t)^2 + \frac{3}{n}\sum_t (\ell_t^{\mathrm{o}}[t]-\ell_t^{*})^2 + \frac{3}{n}\sum_t (\ell_t^{\mathrm{q}}[t]-\ell_t^{\mathrm{o}}[t])^2$$

Error of our solution	Offline learning error	Online learning error	Quantization error

Claim: When the regularization parameter is set as $\gamma_g = \Omega(n^{1/8})$, and the Laplacians L^{q} and L^{o} and normalized, the average error between the online and online quantized HFS predictions decreases at the rate of $O(n^{-1/2})$

$$\frac{1}{n}\sum_t (\ell_t^{\mathrm{q}}[t]-\ell_t^{\mathrm{o}}[t])^2 \le \frac{1}{n}\sum_t \left\|\ell^{\mathrm{q}}[t]-\ell^{\mathrm{o}}[t]\right\|_2^2 \le \frac{n_l}{c_u^2\gamma_g^4}\left\|L^{\mathrm{q}}-L^{\mathrm{o}}\right\|_F^2$$

$$\left\|L^{\mathrm{q}}-L^{\mathrm{o}}\right\|_F^2 \propto O(k^{-2/d})$$

The distortion rate of online k-center clustering is $O(k^{-1/d})$, where d is dimension of the manifold and k is the number of representative vertices

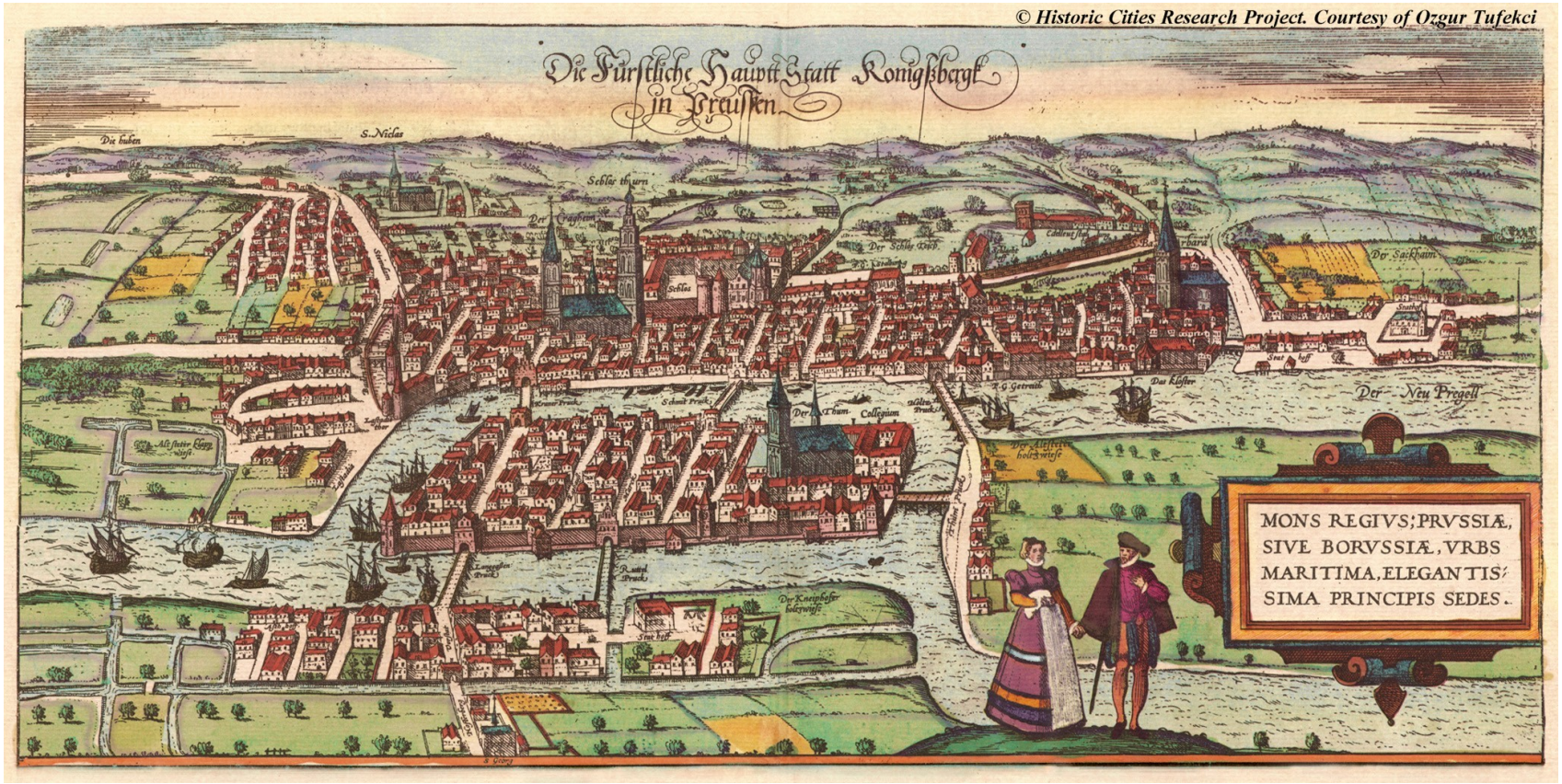
Some of the other topics

- submodularity on graphs
- graph Laplacians, spectral graph theory
- spectral clustering and manifold learning
- social network and recommender systems applications
- semi-supervised learning
- online decision-making on graphs
- learnability on graphs - transductive learning
- generalization bounds by perturbation analysis
- graph neural networks
- graph bandits
- real-world graphs scalability, spectral sparsification

Links to the other courses

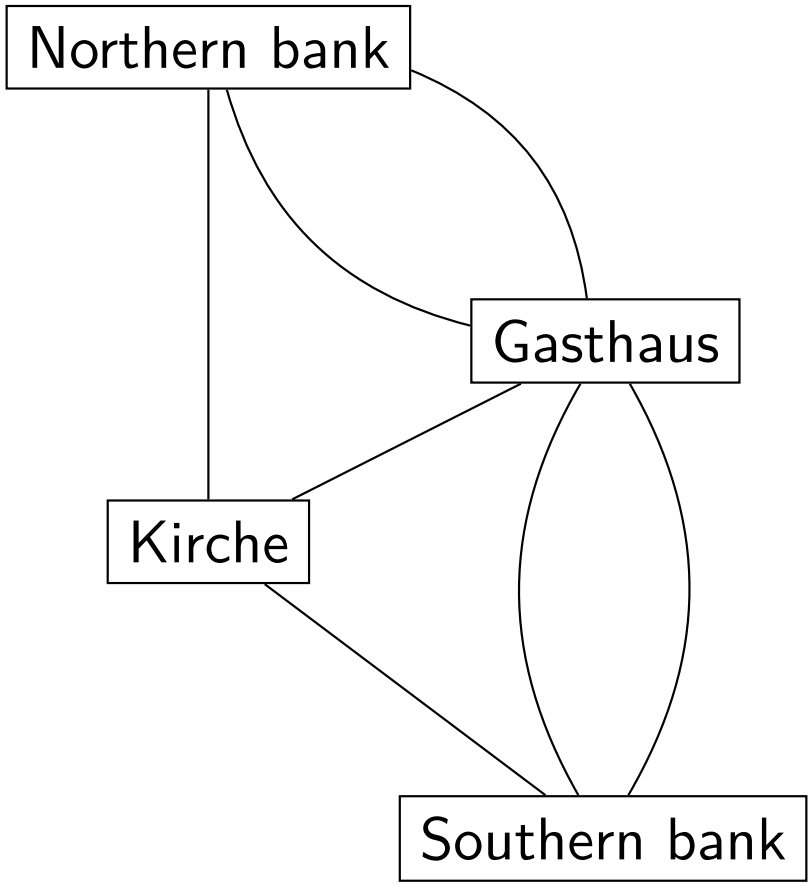
- **Introduction to statistical learning**
  - links to the learning theory on graphs: label propagation, learnability, generalization
- **Reinforcement learning**
  - link to the online learning (bandit) lecture at the end of the semester
- **Advanced learning for text and graph data**
  - data-mining graph course on the topics not covered in this course
  - details on the next slide

Graph theory refresher



Seven bridges of Konigsberg.

Graph theory refresher



Similarity Graphs

Input: $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots, \mathbf{x}_N$

- raw data
- flat data
- vectorial data

Similarity Graphs

Similarity graph: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ — **(un)weighted**

Task 1: For each pair i, j: define a **similarity function** s_{ij}

Task 2: Decide which edges to include

Similarity Graphs

Similarity graph: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ — **(un)weighted**

Task 1: For each pair i, j: define a **similarity function** s_{ij}

Task 2: Decide which edges to include

ε-neighborhood graphs – connect the points with the distances smaller than ε

Similarity Graphs

Similarity graph: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ — **(un)weighted**

Task 1: For each pair i, j: define a **similarity function** s_{ij}

Task 2: Decide which edges to include

ε-neighborhood graphs – connect the points with the distances smaller than ε

k-NN neighborhood graphs – take k nearest neighbors

Similarity Graphs

Similarity graph: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ — **(un)weighted**

Task 1: For each pair i, j: define a **similarity function** s_{ij}

Task 2: Decide which edges to include

ε-neighborhood graphs – connect the points with the distances smaller than ε

k-NN neighborhood graphs – take k nearest neighbors

fully connected graphs - consider everything

Similarity Graphs

Similarity graph: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ — **(un)weighted**

Task 1: For each pair i, j: define a **similarity function** s_{ij}

Task 2: Decide which edges to include

ε-neighborhood graphs – connect the points with the distances smaller than ε

k-NN neighborhood graphs – take k nearest neighbors

fully connected graphs - consider everything

Similarity Graphs

Similarity graph: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ — **(un)weighted**

Task 1: For each pair i, j: define a **similarity function** s_{ij}

Task 2: Decide which edges to include

ε-neighborhood graphs – connect the points with the distances smaller than ε

k-NN neighborhood graphs – take k nearest neighbors

fully connected graphs - consider everything

This is art (not much theory exists).

`http://www.informatik.uni-hamburg.de/ML/contents/people/luxburg/publications/Luxburg07_tutorial.pdf`

Similarity Graphs: ε-neighborhood graphs

Edges connect the points with the distances smaller than ε.

Similarity Graphs: ε-neighborhood graphs

Edges connect the points with the distances smaller than ε.

- distances are roughly on the same scale (ε)

Similarity Graphs: ε-neighborhood graphs

Edges connect the points with the distances smaller than ε.

- distances are roughly on the same scale (ε)
- weights may not bring additional info $\rightarrow$ unweighted

Similarity Graphs: ε-neighborhood graphs

Edges connect the points with the distances smaller than ε.

- distances are roughly on the same scale (ε)
- weights may not bring additional info $\rightarrow$ unweighted
- equivalent to: similarity function is at least ε

Similarity Graphs: ε-neighborhood graphs

Edges connect the points with the distances smaller than ε.

- distances are roughly on the same scale (ε)
- weights may not bring additional info $\rightarrow$ unweighted
- equivalent to: similarity function is at least ε
- theory [Penrose, 1999]: $\varepsilon = ((\log N)/N)^d$ to guarantee connectivity N nodes, d dimension

Similarity Graphs: ε-neighborhood graphs

Edges connect the points with the distances smaller than ε.

- distances are roughly on the same scale (ε)
- weights may not bring additional info $\rightarrow$ unweighted
- equivalent to: similarity function is at least ε
- theory [Penrose, 1999]: $\varepsilon = ((\log N)/N)^d$ to guarantee connectivity N nodes, d dimension
- practice: choose ε as the length of the longest edge in the MST - minimum spanning tree

Similarity Graphs: ε-neighborhood graphs

Edges connect the points with the distances smaller than ε.

- distances are roughly on the same scale (ε)
- weights may not bring additional info $\rightarrow$ unweighted
- equivalent to: similarity function is at least ε
- theory [Penrose, 1999]: $\varepsilon = ((\log N)/N)^d$ to guarantee connectivity N nodes, d dimension
- practice: choose ε as the length of the longest edge in the MST - minimum spanning tree

Similarity Graphs: ε-neighborhood graphs

Edges connect the points with the distances smaller than ε.

- distances are roughly on the same scale (ε)
- weights may not bring additional info $\rightarrow$ unweighted
- equivalent to: similarity function is at least ε
- theory [Penrose, 1999]: $\varepsilon = ((\log N)/N)^d$ to guarantee connectivity N nodes, d dimension
- practice: choose ε as the length of the longest edge in the MST - minimum spanning tree

What could be the problem with this MST approach?

Similarity Graphs: ε-neighborhood graphs

Edges connect the points with the distances smaller than ε.

- distances are roughly on the same scale (ε)
- weights may not bring additional info $\rightarrow$ unweighted
- equivalent to: similarity function is at least ε
- theory [Penrose, 1999]: $\varepsilon = ((\log N)/N)^d$ to guarantee connectivity N nodes, d dimension
- practice: choose ε as the length of the longest edge in the MST - minimum spanning tree

What could be the problem with this MST approach?

Anomalies can make ε too large.

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)
  - option OR: ignore the direction

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)
  - option OR: ignore the direction
  - option AND: include if we have both direction (mutual k-NN)

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)
  - option OR: ignore the direction
  - option AND: include if we have both direction (mutual k-NN)
- how to choose k?

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)
  - option OR: ignore the direction
  - option AND: include if we have both direction (mutual k-NN)
- how to choose k?
- $k \approx \log N$ - suggested by asymptotics (practice: up to $\sqrt{N}$)

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)
  - option OR: ignore the direction
  - option AND: include if we have both direction (mutual k-NN)
- how to choose k?
- $k \approx \log N$ - suggested by asymptotics (practice: up to $\sqrt{N}$)
- for mutual k-NN we need to take larger k

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)
  - option OR: ignore the direction
  - option AND: include if we have both direction (mutual k-NN)
- how to choose k?
- $k \approx \log N$ - suggested by asymptotics (practice: up to $\sqrt{N}$)
- for mutual k-NN we need to take larger k
- mutual k-NN does not connect regions with different density

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)
  - option OR: ignore the direction
  - option AND: include if we have both direction (mutual k-NN)
- how to choose k?
- $k \approx \log N$ - suggested by asymptotics (practice: up to $\sqrt{N}$)
- for mutual k-NN we need to take larger k
- mutual k-NN does not connect regions with different density
- why don't we take $k = N - 1$?

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)
  - option OR: ignore the direction
  - option AND: include if we have both direction (mutual k-NN)
- how to choose k?
- $k \approx \log N$ - suggested by asymptotics (practice: up to $\sqrt{N}$)
- for mutual k-NN we need to take larger k
- mutual k-NN does not connect regions with different density
- why don't we take $k = N - 1$?
  - space and time

Similarity Graphs: k-nearest neighbors graphs

Edges connect each node to its k-nearest neighbors.

- asymmetric (or directed graph)
  - option OR: ignore the direction
  - option AND: include if we have both direction (mutual k-NN)
- how to choose k?
- $k \approx \log N$ - suggested by asymptotics (practice: up to $\sqrt{N}$)
- for mutual k-NN we need to take larger k
- mutual k-NN does not connect regions with different density
- why don't we take $k = N - 1$?
  - space and time
  - manifold considerations (preserving local properties)

Similarity Graphs: Fully connected graphs

Edges connect everything.

Similarity Graphs: Fully connected graphs

Edges connect everything.

- choose a "meaningful" similarity function s

Similarity Graphs: Fully connected graphs

Edges connect everything.

- choose a "meaningful" similarity function s
- default choice:

$$s_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

Similarity Graphs: Fully connected graphs

Edges connect everything.

- choose a "meaningful" similarity function s
- default choice:

$$s_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

- why the exponential decay with the distance?

Similarity Graphs: Fully connected graphs

Edges connect everything.

- choose a "meaningful" similarity function s
- default choice:

$$s_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

- why the exponential decay with the distance?
- σ controls the width of the neighborhoods

Similarity Graphs: Fully connected graphs

Edges connect everything.

- choose a "meaningful" similarity function s
- default choice:

$$s_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

- why the exponential decay with the distance?
- σ controls the width of the neighborhoods
  - similar role as ε

Similarity Graphs: Fully connected graphs

Edges connect everything.

- choose a "meaningful" similarity function s
- default choice:

$$s_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

- why the exponential decay with the distance?
- σ controls the width of the neighborhoods
  - similar role as ε
  - **a** practical rule of thumb: 10% of the average empirical std

Similarity Graphs: Fully connected graphs

Edges connect everything.

- choose a "meaningful" similarity function s
- default choice:

$$s_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

- why the exponential decay with the distance?
- σ controls the width of the neighborhoods
  - similar role as ε
  - **a** practical rule of thumb: 10% of the average empirical std
  - possibility: learn σ_i for each feature independently

Similarity Graphs: Fully connected graphs

Edges connect everything.

- choose a "meaningful" similarity function s
- default choice:

$$s_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

- why the exponential decay with the distance?
- σ controls the width of the neighborhoods
  - similar role as ε
  - **a** practical rule of thumb: 10% of the average empirical std
  - possibility: learn σ_i for each feature independently
- metric learning (a whole field of ML)

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)
  - down-sample

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)
  - down-sample
  - approximate NN

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)
  - down-sample
  - approximate NN
    - **LSH** - Locally Sensitive Hashing

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)
  - down-sample
  - approximate NN
    - **LSH** - Locally Sensitive Hashing
    - **CoverTrees**

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)
  - down-sample
  - approximate NN
    - **LSH** - Locally Sensitive Hashing
    - **CoverTrees**
    - **Spectral sparsifiers**

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)
  - down-sample
  - approximate NN
    - **LSH** - Locally Sensitive Hashing
    - **CoverTrees**
    - **Spectral sparsifiers**
  - sometime we may not need the graph (just the final results)

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)
  - down-sample
  - approximate NN
    - **LSH** - Locally Sensitive Hashing
    - **CoverTrees**
    - **Spectral sparsifiers**
  - sometime we may not need the graph (just the final results)
  - yet another story: when we start with a large graph and want to make it sparse (later in the course)

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)
  - down-sample
  - approximate NN
    - **LSH** - Locally Sensitive Hashing
    - **CoverTrees**
    - **Spectral sparsifiers**
  - sometime we may not need the graph (just the final results)
  - yet another story: when we start with a large graph and want to make it sparse (later in the course)
- these rules have little theoretical underpinning

Similarity Graphs: Important considerations

- *calculate all* s_{ij} *and threshold* has its limits ($N \approx 10000$)
- graph construction step can be a huge bottleneck
- want to go higher? (we often have to)
  - down-sample
  - approximate NN
    - **LSH** - Locally Sensitive Hashing
    - **CoverTrees**
    - **Spectral sparsifiers**
  - sometime we may not need the graph (just the final results)
  - yet another story: when we start with a large graph and want to make it sparse (later in the course)
- these rules have little theoretical underpinning
- similarity is very data-dependent

Daniele Calandriello

dcalandriello@google.com

ENS Paris-Saclay, MVA 2022/2023

https://sites.google.com/view/daniele-calandriello/