Modern Aspects of Unsupervised Learning

Maria-Florina Balcan
Carnegie Mellon University
• Detecting **Overlapping Clusters:** Endogenously Formed Communities
  
  [Balcan, Borgs, Braverman, Chayes, Teng, SODA 2013]

• **Coresets & Distributed Clustering.**
  
  [Balcan-Ehrlich-Liang, NIPS 2013]
  [Balcan-Kanchanapally-Liang-Woodruff, NIPS 2014]
Unsupervised Learning/Clustering

- Extensively studied in many fields.
- Classic goals: output a partition of the data.
Unsupervised Learning/Clustering

• Extensively studied in many fields.
• Classic goals: hierarchical clustering
Overlapping communities

- Social networks
- Professional networks

- Product Purchasing Networks, Citation Networks, Biological Networks, etc
Overlapping communities

Christos Papadimitriou

Colleagues at Berkeley

Databases Systems

Algorithmic Game Theory

TCS
Overlapping communities

Baby's Favorite Songs

lullabies

Electronics

CDs

Kids
Overlapping communities

• Used usually as preprocessing step for data analysis or decision making.

Open Question: rigorous & natural notions; algorithms for finding all of them.

Prior Work:

• Various heuristics and optimization criteria [N’06, K11]
• No general guarantees on # and time needed to find communities meeting natural criteria [MSST07]
Self-Determined Communities in General Affinity Systems
Affinity systems

Basic model (ordinal): \((V=[n], \pi_1, \pi_2, ..., \pi_n)\)

Weighted affinity systems: \((V=[n], a_1, a_2, ..., a_n)\)

\(a_{i,j} \in [0,1]\) - affinity of member \(i\) for member \(j\)

Arise in different areas:

- social sciences
- social networks
- Data mining (e.g., documents, DNA sequences, etc.)
Self Determined Communities in Affinity Systems

Basic model (ordinal): \( (V=\{n\}, \pi_1, \pi_2, \ldots, \pi_n) \)

\( S \subseteq V \) self-determined community if members of \( S \) collectively prefer each other to anyone else outside the community

- \( \# \text{ votes } i \text{ in } S \text{ receives from members in } S \geq \# \text{ of votes } j \text{ not in } S \text{ receives from } S \)
- each \( i \) in \( S \) casts a vote for his \( |S| \) most preferred members

Different communities have different degrees of robustness
**Self Determined Communities in Affinity Systems**

**Definition** \( S \subseteq V (\theta, \alpha, \beta) \) self-determined community if

- if each \( i \) in \( S \) receives \( \geq \alpha |S| \) votes from members of \( S \)
- if each \( j \) not in \( S \) receives \( \leq \beta |S| \) votes from members of \( S \)
- each \( i \) in \( S \) casts a vote for its \( \theta |S| \) most preferred members

**Allows for overlapping communities**

\(|A_1| = n/2 \quad |A_2| = n/2\)

- Each \( s \) in \( A_i \setminus A_j \) ranks elements in \( A_i \) first
- \( A_1, A_2 \) are \((1, 3/4, 1/4)\) self-determined comm.

**Given a set \( S \), easy to efficiently determine whether or not \( S \) is SD.**
Self Determined Communities, Main Results

• A multi-stage approach that leads to a poly time algorithm for finding all communities if \( \theta, \alpha, \beta \) are constant.

• Local procedure: for \( \alpha \geq \frac{1}{2} \), given a random \( v \) in community \( S \), with prob. \( \Omega(2^{\alpha} - 1) \) recovers \( S \) in time \( O(|S| \log|S|) \).

• Weighted affinity systems, Multi-facet affinity systems.

• Connections to \((\alpha, \beta)\) clusters [Mishra, Schreiber, Staton, Tarjan]
  • We prove there exists network with superpoly \# of \((\alpha, \beta)\) clusters and even finding one as hard as hidden clique
Self Determined Communities in Affinity Systems

Theorem

Given \( t = |S| \) find, output a list \( L \) that whp contains \( S \) in time:

\[
\eta^O\left(\frac{\log (1/\gamma)}{\alpha}\right) \left(\frac{\theta \log (1/\gamma)}{\alpha}\right)^O\left(\frac{1}{\gamma^{2}} \log \left(\frac{\theta (1/\gamma)}{\alpha \gamma}\right)\right)
\]

Leads to a poly time algorithm for finding all communities all parameters are constant.
Multi-Stage Approach

Input: Info $I$ about unknown community $S$ (e.g., $|S|$).
Output: List $L$ of subsets of $V$.

**Generate Rough Approximations Step**
- Generate a list $L_1$ of sets $S_1$ s.t. at least one of them is a rough approximation to $S$.

**Purification Step**
- Run a purification procedure to generate a list $L$ s.t. at least one of the elements in $L$ is identical to $S$.

Eliminate from $L$ sets that are not self-determined.
Key Fact \[ k_1 = \log \left( \frac{1}{\gamma} \right) / \alpha \]

\[ \exists \ i_1, \ldots, i_{k-1} \ \text{in} \ S \ \text{s. t. the union of their votes contains} \ \geq 1-\gamma/16 \ \text{fraction of} \ S. \]

Proof

Any \( \tilde{S} \subseteq S \) the receives at least \( \alpha |S| \ |\tilde{S}| \) votes from \( S \), so \( \exists i_{\tilde{S}} \in S \) that votes for at least \( \alpha \ |\tilde{S}| \) members of \( \tilde{S} \).

The existence of \( i_1, \ldots, i_M \) proven in a greedy fashion.
Self Determined Communities in Affinity Systems

**Key Fact** \( k_1 = \log(1/\gamma)/\alpha \)

\( \exists i_1, \ldots, i_{k-1} \) in \( S \) s. t. the union of their votes contains \( \geq 1-\gamma/16 \) fraction of \( S \).

\( |S_1| \leq k_1 \theta t, |S|=t \)

---

**Generate Rough Approximations Step**

- Search over all sets \( U, |U| = k_1 \). For each \( U \), let \( S_1 \) be the set of elements voted by \( U \); add \( S_1 \) to \( L_1 \).
Self Determined Communities in Affinity Systems

**Key Fact** \( k_2 = O(\log(16\theta k_1/\delta \gamma)/\gamma^2) \)

If draw \( U_2 \) a set of \( k_2 \) pts at random from \( S \cap S_1 \), consider \( S_2 \) set voted by \((\alpha - \gamma/2)\) fraction of \( U_2 \), then whp

\[ |\Delta (S_2, S)| \leq \gamma \frac{t}{8}. \]

**Purification Step** \( N_2 = O((\theta k_1)^{k_2}\log(1/\delta)) \)

For each \( S_1 \) to \( L_1 \), repeat \( N_2 \) times

- Pick \( k_2 \) points at random from \( S_1 \) (get \( U_2 \)) and let \( S_2 \) be the set voted by \((\alpha - \gamma/2)\) fraction of \( U_2 \).
- Let \( S_3 \) be the set voted by \((\alpha - \gamma/2)\) fraction of \( S_2 \).
Theorem

Given $t = |S|$ find, output a list $L$ that whp contains $S$ in time:

$$n^{O\left(\frac{\log (1/\gamma)}{\alpha}\right)} \left(\frac{\theta \log (1/\gamma)}{\alpha}\right)^O\left(\frac{1}{\gamma^2 \log \left(\frac{\theta (1/\gamma)}{\alpha \gamma}\right)}\right)$$

Leads to a poly time algorithm for finding all communities all parameters are constant.
Local Procedure

**Theorem:** For \( \alpha \geq \frac{1}{2} \), given a random \( v \) in community \( S \), with prob. \( \Omega(2\alpha - 1) \) recovers \( S \) in time \( O(|S| \log|S|) \).

- Similar multi-stage approach; main challenge, providing a local version for rough approxs.

**Note:** not possible to start from any seed vertex \( v \) in \( S \).
  
  - e.g., if \( v \) is voted first by everyone in \( V \)

- We show that a constant fraction of the nodes in \( S \) are sufficiently “representative” of \( S \) to enable recovering \( S \)
Local Procedure

**Key Fact** \( \eta = 2^{\alpha - 1}, \exists T \subseteq S, |T| \geq \eta \) s.t. for \( v \in T \) and \( u \in S \),

\[
\Pr[R(R(v)) = u] \geq \frac{(\alpha - 1/2)/\theta^2}{t}
\]

**Generate Rough Approximations** Given \( v \in T \):

- Compute \( R(R(v)) \) for \( O((1/p)\log t) \) times.
- \( S_1 = \) all \( u \) hit at least a \( c\log t \) times.

- Whp \( S_1 \) includes all of \( S \), and in total of \( O(t) \) points.
Conclusions

• Natural notion of self-determined community.

• A poly time algorithm for finding all communities if $\theta, \alpha, \beta$ are constant; stronger guarantees for $\alpha \geq \frac{1}{2}$.

Open Questions

• Input affinity system is typically only a projection of the true underlying affinity system.

• Interactive community detection.
Many ML problems today involve massive amounts of data distributed across multiple locations.

Often would like a good clustering over union of datasets.
Distributed Clustering

Data distributed across multiple locations.

E.g., medical data
Distributed Clustering

Data distributed across multiple locations.

E.g., scientific data
Distributed Clustering

• Data distributed across multiple locations.
• Each has a piece of the overall data pie.
• To cluster the overall data, must communicate.

Important question: how much communication?
Plus, privacy & incentives.
Center Based Clustering

k-median: find center pts $c_1, c_2, \ldots, c_k$ to minimize $\sum_x \min_i d(x, c_i)$

k-means: find center pts $c_1, c_2, \ldots, c_k$ to minimize $\sum_x \min_i d^2(x, c_i)$

Key idea: use coresets.

Coresets short summaries capturing relevant info w.r.t. all clusterings.

**Def**: An $\epsilon$-coreset for a set of pts $S$ is a set of points $\tilde{S}$ s.t. and weights $w$: $\tilde{S} \rightarrow R$ s.t. for any sets of centers $c$:

$$
(1 - \epsilon)\text{cost}(S, c) \leq \sum_{p \in D} w_p \text{cost}(p, c) \leq (1 + \epsilon)\text{cost}(S, c)
$$

Algorithm (centralized)

- Find a coreset $\tilde{S}$ of $S$. Run an approx. algorithm on $\tilde{S}$.
Distributed Clustering [Balcan-Ehrlich-Liang, NIPS 2013]

k-median: find center pts $c_1, c_2, \ldots, c_k$ to minimize $\sum_x \min_i d(x, c_i)$

k-means: find center pts $c_1, c_2, \ldots, c_k$ to minimize $\sum_x \min_i d^2(x, c_i)$

• Key idea: use coresets, short summaries capturing relevant info w.r.t. all clusterings.

• [Feldman-Langberg STOC’11] show that in centralized setting one can construct a coreset of size $O(kd/\epsilon^2)$

• By combining local coresets, get a global coreset; the size goes up multiplicatively by $s$.

• In [Balcan-Ehrlich-Liang, NIPS 2013] show a two round procedure with communication only $O(kd/\epsilon^2 + sk)$

[As opposed to $O(s kd/\epsilon^2)$]
Clustering, Coresets  [Feldman-Langberg’11]

[FL’11] construct in centralized cases a coreset of size $O(kd/\epsilon^2)$.

1. Find a constant factor approx. $B$, add its centers to coreset
   [this is already a very coarse coreset]
2. Sample $O(kd/\epsilon^2)$ pts according to their contribution to the
   cost of that approximate clustering $B$.

Key idea: one way to think about this construction

- Upper bound penalty we pay for $p$ under any set of centers $c$
  by distance between $p$ and its closest center $b_p$ in $B$
  - For any set of centers $c$, penalty we pay for point $p$
    $$f(p) = \text{cost}(p, c) - \text{cost}(b_p, c)$$
  - Note $f(p) \in [-\text{cost}(p, b_p), \text{cost}(p, b_p)]$.

  This motivates sampling according to $\text{cost}(p, b_p)$
Distributed Clustering [Balcan-Ehrlich-Liang, NIPS 2013]

Feldman-Langberg’11 show that in centralized setting one can construct a coreset of size $O(kd/\epsilon^2)$.

**Key idea:** in distributed case, show how to do this using only local constant factor approx.

1. Each player, finds a local constant factor approx. $B_i$ and sends $\text{cost}(B_i, P_i)$ and the centers to the center.
2. Center sample $n = O(kd/\epsilon^2)$ pts $n = n_1 + \cdots + n_s$ from multinomial given by these costs.
3. Each player $i$ sends $n_i$ points from $P_i$ sampled according to their contribution to the local approx.
Distributed Clustering [Balcan-Ehrlich-Liang, NIPS 2013]

\[ \text{k-means: find center pts } c_1, c_2, \ldots, c_k \text{ to minimize } \sum_x \min_i d^2(x, c_i) \]
Open questions

• Efficient algorithms in noisy settings; handle failures, delays.

• Even better dependence on $1/\epsilon$ for communication efficiency for clustering via boosting style ideas.
  
  • Can use distributed dimensionality reduction to reduce dependence on $d$.  
    [Balcan-Kanchanapally-Liang-Woodruff, NIPS 2014]

• More refined trade-offs between communication complexity and computational complexity.