Advanced Topics in Machine Learning and NLP
Graph Clustering

Luca Zanetti

email: luca.zanetti@cl.cam.ac.uk
What is clustering?

Clustering is the task of dividing objects in groups so that similar objects are grouped together and dissimilar objects are separated in different groups.

This is not a very precise definition... but hard to find a much better one!
On the hardness of clustering

Clustering is hard:

- Unsupervised learning technique
  - we cannot learn from our mistakes
  - there’s no “ground truth”

- Inherently difficult to formalise:
  - Cluster sharing is an equivalence relation, but the notion of similarity is not transitive
  - “Impossibility theorem for clustering” (Kleinberg, NIPS’13)

- Most formalisations are actually NP-hard

However:

- We have algorithms that “work in practice"
- The more well-clustered the data, the better the quality of the clustering produced
Different notions of clustering

Different formalisations for different domains/applications, e.g.:

- Geometric clustering: partition points in a Euclidean space
  - $k$-means, $k$-medians, $k$-centres, etc.

- Graph clustering: partition vertices in a graph
  - modularity, conductance, min-cut, etc.
Graph clustering

Partition the graph into pieces (clusters) so that vertices in the same piece have, on average, more connections among each other than with vertices in other clusters.
Why study graph clustering?

- Many practical applications, e.g.:
  - Community detection
  - Group webpages according to their topics
  - Find proteins performing the same function within a cell
  - Image segmentation
  - Identify bottlenecks in a network

- Connections with different areas of mathematics and TCS, e.g.:
  - Random walk theory
  - Combinatorics
  - Theory of metric spaces
  - Approximation algorithms
  - Complexity theory
Let’s try to be a bit more formal
Weighted graphs and their adjacency matrix

\[ G = (V, E, w) \] with weight function \( w \), s.t.

- \( w: V \times V \rightarrow \mathbb{R}_{\geq 0} \)
- \( w(u, v) > 0 \iff \{u, v\} \in E \)
- \( w(u, v) = w(v, u) \)

The adjacency matrix of \( G \) is the \( n \) by \( n \) matrix \( A \) defined as

\[ A_{u,v} = w(u, v) \]

\[
A = \begin{pmatrix}
0 & 16 & 0 & 9 \\
16 & 0 & 9 & 0 \\
0 & 9 & 0 & 7 \\
9 & 0 & 7 & 0
\end{pmatrix}
\]
Graph conductance

Let $G = (V, E, w)$ and $\emptyset \neq S \subset V$. The conductance (edge expansion) of $S$ is

$$\phi(S) := \frac{w(S, V \setminus S)}{\min\{\text{vol}(S), \text{vol}(V \setminus S)\}}$$

where $w(S, V \setminus S) = \sum_{u \in S, v \notin S} w(u, v)$ and $\text{vol}(S) = \sum_{u \in S} d_u$.

The conductance of $G$ is

$$\phi(G) := \min_{\emptyset \neq S \subset V} \phi(S)$$

NP-hard to compute!

- $\phi(S) = \frac{5}{9}$
- $\phi(G) \in [0, 1]$ and $\phi(G) = 0$ iff $G$ is disconnected
- If $G$ is a complete graph, then $|E(S, V \setminus S)| = |S| \cdot (n - |S|)$ and $\phi(G) \approx 1/2$. 
A (very brief) primer on Spectral Graph Theory
Graphs and matrices

Graphs

1 -- 2

3 -- 4

- Connectivity
- Bipartiteness
- Graph partitioning
- Number of triangles
- Graph isomorphism
- Max-flow
- ...

Matrices

\[
\begin{pmatrix}
0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0
\end{pmatrix}
\]

- Eigenvalues
- Eigenvectors
- Inverse
- Matrix-powers
- ...

R250: Graph Clustering
Adjacency matrix

Let $G = (V, E)$ be an undirected graph. The adjacency matrix of $G$ is the $n$ by $n$ matrix $A$ defined as

$$A_{u,v} = \begin{cases} 1 & \text{if } \{u, v\} \in E \\
0 & \text{otherwise.} \end{cases}$$

Properties of $A$:

- The sum of elements in each row/column $i$ equals the degree of the corresponding vertex $i$, $\deg(i)$
- Since $G$ is undirected, $A$ is symmetric
Let $G = (V, E)$ be a $d$-regular undirected graph. The (normalised) Laplacian matrix of $G$ is the $n$ by $n$ matrix $L$ defined as

$$L = I - \frac{1}{d} A,$$

where $I$ is the $n \times n$ identity matrix.

Properties of $L$:
- The sum of elements in each row/column equals zero
- $L$ is symmetric
Eigenvalues and Graph Spectrum

Eigenvalues and eigenvectors

Let $M \in \mathbb{R}^{n \times n}$, $\lambda \in \mathbb{C}$ is an eigenvalue of $M$ if and only if there exists $x \in \mathbb{R}^n \setminus \{0\}$ such that

$$Mx = \lambda x.$$ 

We call $x$ an eigenvector of $M$ corresponding to the eigenvalue $\lambda$.

Graph Spectrum

Let $L$ be the Laplacian matrix of a $d$-regular graph $G$ with $n$ vertices. Then, $L$ has $n$ real eigenvalues $\lambda_1 \leq \cdots \leq \lambda_n$ and $n$ corresponding orthonormal eigenvectors $f_1, \ldots, f_n$. 
Let $M$ be an $n$ by $n$ symmetric matrix with eigenvalues $\lambda_1 \leq \cdots \leq \lambda_n$. Then,

$$\lambda_k = \min_{x^{(1)}, \ldots, x^{(k)} \in \mathbb{R}^n \setminus \{0\}, \ i \in \{1, \ldots, k\}} \max_{x^{(i)} \perp x^{(j)}} \frac{x^{(i)^T M x^{(i)}}}{x^{(i)^T x^{(i)}}}.$$

The eigenvectors corresponding to $\lambda_1, \lambda_2, \ldots, \lambda_k$ minimise such expression.

$$\lambda_1 = \min_{x \in \mathbb{R}^n \setminus \{0\}} \frac{x^T M x}{x^T x}, \quad \lambda_2 = \min_{x \perp f_1 \in \mathbb{R}^n \setminus \{0\}} \frac{x^T M x}{x^T x}$$

minimised by an eigenvector $f_1$ for $\lambda_1$ minimised by $f_2$
Quadratic forms of the Laplacian

Lemma

Let $L$ be the Laplacian matrix of a $d$-regular graph $G = (V, E)$ with $n$ vertices. For any $x \in \mathbb{R}^n$,

$$x^T L x = \sum_{\{u,v\} \in E} \frac{(x_u - x_v)^2}{d}.$$

Proof:

$$x^T L x = x^T \left( I - \frac{1}{d} A \right) x = x^T x - \frac{1}{d} x^T A x$$

$$= \sum_{u \in V} x_u^2 - \frac{2}{d} \sum_{\{u,v\} \in E} x_u x_v$$

$$= \frac{1}{d} \sum_{\{u,v\} \in E} (x_u^2 + x_v^2 - 2x_u x_v)$$

$$= \sum_{\{u,v\} \in E} \frac{(x_u - x_v)^2}{d}.$$
Useful facts of graph spectra

Lemma

Let \( L \) be the Laplacian matrix of \( G = (V, E) \) with eigenvalues \( \lambda_1 \leq \cdots \leq \lambda_n \).

1. \( \lambda_1 = 0 \) with eigenvector \( 1 \)
2. the multiplicity of the eigenvalue 0 is equal to the number of connected components in \( G \)
3. \( \lambda_n \leq 2 \)
4. \( \lambda_n = 2 \) iff there exists a bipartite connected component.

Proof of 1:

- \( \lambda_1 = \min_{x \in \mathbb{R}^n \setminus \{0\}} \frac{x^T L x}{x^T x} \) by Courant-Fisher
- \( x^T L x = \sum_{\{u,v\} \in E} \frac{(x_u - x_v)^2}{d} \geq 0 \); \( x^T x = \sum_v x_v^2 \geq 0 \implies \lambda_1 \geq 0 \)
- \( 1^T L 1 = 0 \implies \lambda_1 \leq 0 \)
- \( \implies \lambda_1 = 0 \).
Proof sketch of 2 (multiplicity of 0 equals the no. of connected components):

1. \((\implies)\) We want to prove:
   \(G\) has exactly \(k\) connected components \(C_1, \ldots, C_k \subseteq V\)
   \(\implies \lambda_k = 0, \lambda_{k+1} > 0\)

2. Take \(\chi_{C_i} \in \{0, 1\}^n\) such that \(\chi_{C_i}(u) = 1 \iff u \in C_i\)

3. \(\chi_{C_i}^T L \chi_{C_i} = \frac{1}{d} \cdot \sum_{\{u,v\} \in E} (\chi_{C_i}(u) - \chi_{C_i}(v))^2 = 0\)
   \(\implies \lambda_k = 0\)

4. For \(\lambda_{k+1} = 0\) we would need \(k + 1\) orthogonal vectors constant on the \(k\) connected components of \(G\): this is impossible \(\implies \lambda_{k+1} > 0\)

5. \((\impliedby)\) we want to prove that if \(0 = \lambda_k < \lambda_{k+1}\), \(G\) has exactly \(k\) connected components

6. there exist \(f_1, \ldots, f_k\) orthonormal such that \(\sum_{\{u,v\} \in E} (f_i(u) - f_i(v))^2 = 0\)

7. \(\implies f_1, \ldots, f_k\) constant on connected components

8. since \(f_1, \ldots, f_k\) are pairwise orthogonal, \(G\) must have \(k\) different connected components.

9. if \(G\) had more than \(k\) connected components, \(\lambda_{k+1} = 0\)
   \(\implies G\) has exactly \(k\) connected components
Eigenvalues and graph clustering

Lemma

Let $L$ be the Laplacian matrix of $G$. $L$ has eigenvalue 0 with multiplicity equal to the number of connected components of $G$.

- $G$ has $k$ “perfect clusters” (disconnected subsets) $\iff \lambda_k = 0$.
- Is there a “robust” version of this fact?
- Yes! $\lambda_k$ is small $\iff$ there are $k$ loosely connected clusters.
$\lambda_2$ and Combinatorial Expansion on Concrete Networks

1D Grid

$\lambda_2 \sim n^{-2}$
$\phi \sim n^{-1}$

2D Grid

$\lambda_2 \sim n^{-1}$
$\phi \sim n^{-1/2}$

3D Grid

$\lambda_2 \sim n^{-2/3}$
$\phi \sim n^{-1/3}$

Hypercube

$\lambda_2 \sim (\log n)^{-1}$
$\phi \sim (\log n)^{-1}$

Random Graph (Expanders)

$\lambda_2 = \Theta(1)$
$\phi = \Theta(1)$

Binary Tree

$\lambda_2 \sim n^{-1}$
$\phi \sim n^{-1}$
Relating Spectral and Combinatorial Expansion

**Cheeger’s inequality**

Let $G$ be a graph and $\lambda_1 \leq \cdots \leq \lambda_n$ be the eigenvalues of its Laplacian matrix. Then,

$$\frac{\lambda_2}{2} \leq \phi(G) \leq \sqrt{2\lambda_2}.$$

**Spectral partitioning:**

1. Let $x$ be the eigenvector corresponding to $\lambda_2$ and $y = D^{-1/2}x$.
2. Order the vertices so that $y_1 \leq y_2 \leq \cdots \leq y_n$ (embed $V$ on $\mathbb{R}$)
3. Try all $n-1$ sweep cuts of the form ($\{1, 2, \ldots, k\}, \{k + 1, \ldots, n\}$) and return the one with smallest conductance

- It returns $S \subset V$ such that $\phi(S) \leq \sqrt{2\lambda_2} \leq 2\sqrt{\phi(G)}$
- no constant factor approximation (in the worst case)
- $\lambda_2$ can be thought as a convex relaxation for $\phi(G)$
How can we find more than 2 clusters?
**k-means clustering**

**INPUT:**
- a set of \( n \) points \( X = \{x_1, \ldots, x_n\} \in \mathbb{R}^d \)
- the number of clusters \( k \geq 2 \)

**GOAL:**
- assign the points to \( k \) clusters so as to minimise the intra-cluster variance:

\[
\arg\min_{S_1, \ldots, S_k \text{ partition of } X} \sum_{i=1}^k \sum_{x \in S_i} \|x - c(S_i)\|^2
\]

where

\[
c(S_i) = \frac{1}{|S_i|} \sum_{x \in S_i} x \quad \text{is the center of } S_i
\]

- \( k \)-means clustering is **NP-hard**!
- there are good approximation algorithms
- simple heuristics (usually!) work well in practice
$k$-means clustering (examples)
Non-regular weighted graphs

Let $G = (V, E, w)$ be an undirected weighted graph, i.e.,

- $w: V \times V \rightarrow \mathbb{R}_{\geq 0}$
- $w(u, v) > 0 \iff \{u, v\} \in E$
- $w(u, v) = w(v, u)$

The (normalised) Laplacian matrix of $G$ is the $n$ by $n$ matrix $L$ defined as

$$L = I - D^{-1/2}AD^{-1/2}$$

where $D$ is a diagonal $n \times n$ matrix such that $D_{uu} = \text{deg}(u)$, where $\text{deg}(u) = \sum_{v: \{u, v\} \in E} w(u, v)$, and $A$ is the weighted adjacency matrix of $G$, i.e., $A_{uv} = w(u, v)$

- $L$ is symmetric
- Most spectral properties are preserved (e.g., $\lambda_k = 0 \implies k$ connected components) but $D^{1/2}1$ is the eigenvector of eigenvalue 0
Non-regular weighted graphs (2)

\[ L = I - D^{-1/2}AD^{-1/2} \]

\[
\begin{pmatrix}
1 & -\frac{3}{\sqrt{12}} & 0 & 0 & 0 \\
-\frac{3}{\sqrt{12}} & 1 & -\frac{1}{4} & 0 & 0 \\
0 & -\frac{1}{4} & 1 & -\frac{3}{\sqrt{20}} & 0 \\
0 & 0 & -\frac{3}{\sqrt{20}} & 1 & -\frac{2}{\sqrt{10}} \\
0 & 0 & 0 & -\frac{2}{\sqrt{10}} & 1
\end{pmatrix}
\]
**Spectral clustering**

**Goal:** Partition $G = (V, E, w)$ in $k \geq 2$ well-separated clusters

$f_1, \ldots, f_k$ eigenvectors of the Laplacian of $G$ (corresponding to $\lambda_1, \ldots, \lambda_k$)

1. Compute the spectral embedding $F: V \rightarrow \mathbb{R}^k$

   $$F(u) = \frac{1}{\sqrt{\text{deg}(u)}} (f_1(u), \ldots, f_k(u))^T$$

2. Solve $k$-means on $\{F(u)\}_{u \in V}$

3. Partition $G$ according to the output of $k$-means
Example: Stochastic Block Models

Graph $G = (V, E)$ with clusters $S_1, S_2, S_3 \subset V$; \(0 \leq q < p \leq 1\)

\[
\Pr[u \sim v] = \begin{cases} 
p & u, v \in S_i 
q & u \in S_i, v \in S_j, i \neq j
\end{cases}
\]

$|V| = 300, |S_i| = 100$

$p = 0.08, q = 0.01.$
Example: US migration data

- Consider a dataset regarding internal migration in the US.
- For each pair of counties \((i, j)\), \(M(i, j)\) represents the number of people who migrated from \(i\) to \(j\) in the timeframe 2000-2010.
- This can be seen as a weighted directed graph, where each node is a county and \(M\) is its weighted adjacency matrix.
- We first make this graph undirected: compute \(M + M^T\)
- We apply Spectral Clustering with \(k = 10\)
Example: England+Wales migration data

- For each pair of local authorities \((i, j)\), \(M(i, j)\) represents the number of people who migrated from \(i\) to \(j\) in the timeframe 2012-2017.
- We first make the graph undirected: \(M + M^T\), and then compute its random walk matrix.
- We apply Spectral Clustering \((k = 8)\).
Spectral clustering beyond graphs
$k$-means clustering (examples)
Why \( k \)-means fail

\( k \)-means is able to recover only convex clusters:

- it divides the space in \( k \) regions with the following property: if we connect two points belonging to the same region, we never intersect any other region
Similarity graph

Given $X = \{x_1, \ldots, x_n\} \in \mathbb{R}^d$, construct $G = (V, E, w)$:

- $x_i \in X \mapsto v_i \in V$
- $E = \binom{V}{2}$
- $w(v_i, v_j) = \exp \left( -\frac{||x_i - x_j||^2}{2\sigma^2} \right)$ (Gaussian similarity function)

Remarks:

- $w(v_i, v_j)$ is large if $x_i$ is close to $x_j$
- value of $\sigma \geq 0$ depends on the application (choose it by trial and error)
- large $\sigma$ if, on average, pairwise nearest neighbours are far apart

Problem: Since $G$ is complete, from $\Theta(dn)$ to $\Theta(n^2)$ space.

Possible solution: $r$-nearest neighbour graph ($v_i \sim v_j$ iff $x_j$ is one of the $r$-nearest neighbours of $x_i$ or vice versa)

From geometric to graph clustering!
Example

Similarity graph: Gaussian with $\sigma = 0.1$. Only edges with weight $\geq 0.01$ shown.

Spectral partitioning:

1. Compute the eigenvector $f_2$ corresponding to $\lambda_2$
2. Order the vertices so that $f_2(u_1) \leq f_2(u_2) \leq \cdots \leq f_2(u_n)$
3. Choose “sweep” cut $(\{u_1, \ldots, u_i\}, \{u_{i+1}, \ldots, u_n\})$ with smallest conductance
References

Ulrike von Luxburg
A tutorial on spectral clustering
Statistics and computing (2007)

Santo Fortunato
Community detection in graphs
Physics reports (2010)

Daniel A. Spielman
Spectral Partitiong in a Stochastic Block Model
Lecture notes for Spectral Graph Theory (2015)
Appendix A: image segmentation

**Goal:** identify different objects in an image

Construct similarity graph as follows:
- A pixel $p$ is characterised by its position in the image and by its RGB value
- map pixel $p$ in position $(x, y)$ to a vector $v_p = (x, y, r, g, b)$
- construct similarity graph as explained earlier

Original image

Output SC (Gaussian, $\sigma = 10$)
Appendix B: Lloyd’s algorithm for $k$-means

**INPUT:** $X \subset \mathbb{R}^d$, $k \geq 2$

**GOAL:**

$$\arg\min_{S_1, \ldots, S_k \text{ partition of } X} \sum_{i=1}^{k} \sum_{x \in S_i} \|x - c(S_i)\|^2$$

where $c(S_i) = \frac{1}{|S_i|} \sum_{x \in S_i} x$

Algorithm:

1. choose $k$ random candidate centres $c_1, \ldots, c_k \in \mathbb{R}^d$
2. form clusters $S_1, \ldots, S_k$ by assigning each $x \in X$ to its nearest centre $c_j$: 
   $$S_j = \{x \in X : j = \min_{1 \leq i \leq k} \|x - c_i\|^2\}$$
3. compute the new centres of the clusters: $c_j = \frac{1}{|S_j|} \sum_{x \in S_j} x$
4. Repeat steps 2-3 until clusters don’t change anymore.

- work usually well in practice, but
- exponential time to converge in the worst case
- no approximation guarantee
- by cleverly choosing the initial centres, we can obtain a $O(\log k)$-approximation algorithm ($k$-means++)