Newman Girvan Method: Centrality-based Splitting + Modularity

Last example of this part: bringing it all together (see [3]):


1. Calculate edge betweenness for all edges
2. Remove edge with highest edge betweenness

- Use Modularity as intra cluster coherence (f) cluster validity measure (g=0) to optimally cut dendrogram:

$$Q = \sum_i (e_{ii} - \alpha_i^2) = \text{Tr} e - \| e^2 \|$$
Newman Girvan Method: Centrality-based Splitting + Modularity

Modularity:

- **k clusters** → **k x k symmetric matrix** $\mathbf{e}$: $e_{ij} = \frac{|E(C_i \cap C_j)|}{|E|}$: fraction of edges between communities
  - $\text{Tr } \mathbf{e} = \sum_i e_{ii}$: fraction of edges within communities
  - $a_i = \sum_j e_{ij}$: fraction of edges that connect to cluster $C_i$

- **Random network (keep $a_i$ fixed)**: $e_{ij}^{\text{rand}} = a_i a_j \rightarrow e_{ii}^{\text{rand}} = a_i^2$

- **f: Compare (difference)** real with rnd: $Q = \sum_i (e_{ii} - a_i^2) = \text{Tr } \mathbf{e} - \| \mathbf{e}^2 \|$
Modularity:
- $k$ clusters $\rightarrow k \times k$ symmetric matrix $e$: $e_{ij} = |E(C_i,C_j)| / |E|$ : fraction of edges between communities
- $\sum e_{ii} :$ fraction of edges within communities
- $a_i = \sum e_{ij} :$ fraction of edges that connect to cluster $C_i$
- Random network (keep $a_i$ fixed): $e_{ij}^{\text{rd}} = a_ia_j \rightarrow e_{ii}^{\text{rd}} = a_i^2$
- f: Compare (difference) real with rnd $\rightarrow Q = \sum (e_{ii} - a_i^2) = \text{Tr} e - \| e^2 \|$

Modularity:
- $k$ clusters $\rightarrow k \times k$ symmetric matrix $e$: $e_{ij} = |E(C_i,C_j)| / |E|$ : fraction of edges between communities
- $\sum e_{ii} :$ fraction of edges within communities
- $a_i = \sum e_{ij} :$ fraction of edges that connect to cluster $C_i$
- Random network (keep $a_i$ fixed): $e_{ij}^{\text{rd}} = a_ia_j \rightarrow e_{ii}^{\text{rd}} = a_i^2$
- f: Compare (difference) real with rnd $\rightarrow Q = \sum (e_{ii} - a_i^2) = \text{Tr} e - \| e^2 \|$
Newman Girvan Method: Centrality-based Splitting + Modularity

Modularity:

- $k$ clusters $\rightarrow k \times k$ symmetric matrix $e$: $e_{ij} = \frac{|E(C_i, C_j)|}{|E|}$: fraction of edges between communities

- $\sum_i e_{ii}$: fraction of edges within communities

- $\sum_j e_{ij}$: fraction of edges that connect to cluster $C_i$

- Random network (keep $a_i$ fixed): $e_{ij}^{md} = a_i a_j \rightarrow e_{ii}^{md} = a_i^2$

- $Q = \sum_i (e_{ii} - a_i^2) = \text{Tr e} - \| e^2 \|$

- Random network (keep $a_i$ fixed): $e_{ij}^{md} = a_i a_j \rightarrow e_{ii}^{md} = a_i^2$

- $Q = \sum_i (e_{ii} - a_i^2) = \text{Tr e} - \| e^2 \|$

Newman Girvan Method: Centrality-based Splitting + Modularity

Modularity:

- In [1]: different notion (not keeping $a_i$ fixed): $\sum_{i=1}^k \left( |E(C_i)| - \frac{|G_i| \cdot (|G_i| - 1)}{n \cdot (n-1)} \right)$

- In [4]: Newman's version for weighted graphs: idea: use multiple edges to model weights

Data Mining: Metric Clustering

\[
\begin{array}{c|cccc}
& A & B & C & D \\
\hline
A & 0 & 1 & 3 & 1 \\
B & 1 & 0 & 2 & 1 \\
C & 3 & 2 & 0 & 0 \\
D & 1 & 1 & 0 & 0 \\
\end{array}
\]
Finding Clusters in Profiles

- **Node profiles** may contain:
  - Personal data:
    - name, age, sex, role-description, skill-description etc.,
    - contextual personal data
  - location, velocity, current emotional state etc.

- **Edge profiles** may contain:
  - Averaged data:
    - general type of relation, average strength of relation, etc.
  - Time-dependent or contextual data:
    - record of relation instantiations (with time & space information), momentary state of relation (weight, state of instantiation, etc.) etc.

Examples for profile elements that can be embedded in metric spaces:

- **Location & Velocity**: Metric space: \( (\mathbb{R}^3, || \cdot ||) \)

- **Text describing Interests**: Metric space: \( (\mathbb{R}^{\text{Voc}}, || \cdot ||) \) where Voc denotes the Vocabulary of the text.

Examples:

- **I like to dance samba, bake pizza, watch tv and plant trees in the garden. I also like to bake cakes.**

Often: instead of term-frequency (tf) alone: use term-frequency * inverse document frequency (idf); \( \text{idf} = \log (\# \text{of docs where} t \text{ occurs} / \# \text{of docs}) \)

How do we compute clusters in metric spaces?

- **Group models**: How do we compute socially meaningful clusters in metric spaces (and thus avoid quasi-groups)?

First some notations / basics:

- In graph clustering we had: A graph clustering \( C = \{C_1, C_2, ..., C_K\} \) is a partition of \( V \) into non-empty subsets \( C_k \)

- Now: clustering \( \mathcal{C} : \mathcal{X} \rightarrow \mathcal{I} \) : mapping of a metric value space \( \mathcal{X} \) to a set of cluster indices \( \mathcal{I} \)

Clusterings can be:

- exclusive or non-exclusive
- crisp or fuzzy
- hierarchical or non-hierarchical
Metric variant of Single / Complete link clustering

- Metric variant of Single / Complete link clustering: Hierarchical, crisp, non-overlapping

- Completely analogous to graph clustering case: Start with singletons and on each level of the dendrogram merge two clusters with minimal distance (cost)
  - Single link:
    $d(C_k_1, C_k_2) = \min_{\{n_1, n_2| x_{n_1} \in C_k_1 \land x_{n_2} \in C_k_2\}} \|x_{n_1} - x_{n_2}\|
  
  - Complete link:
    $d(C_k_1, C_k_2) = \max_{\{n_1, n_2| x_{n_1} \in C_k_1 \land x_{n_2} \in C_k_2\}} \|x_{n_1} - x_{n_2}\|$

K-Means Clustering

- General idea (also valid in graph clustering): Optimize objective function that formalizes clustering paradigm.

- K-Means: Optimize intra cluster coherence:
  - Describe cluster C_k by prototype \(\mu_k\); prototype need not be an actual pattern (If so, algorithm works with slight modifications as well)
  - Determine cluster for each pattern x_n by nearest neighbour rule:
    $$C(x_n) = k_a \leftrightarrow \|x_n - \mu_{k_a}\| = \min_k \|x_n - \mu_k\|$$
K-Means: Optimize intra cluster coherence:

- Find prototypes by optimizing objective function modeling intra cluster coherence as mean square error

\[
J_{SQE} = \sum_{k=1}^{K} \sum_{\{n \mid x_n \in C_k\}} \|x_n - \mu^k\|^2
\]

\[
\frac{dJ_{SQE}}{d\mu^k} = 0 \implies \mu^k = \frac{1}{|C_k|} \sum_{\{n \mid x_n \in C_k\}} x_n
\]

- Cluster prototypes are barycenters ("centers of gravity") of their clusters.
**K-Means Clustering**

- Interesting aspect: How do we determine correct number k of clusters? (Same problem with graph clustering: where to cut dendrogram?)
- Answer: Compute for every k clusterings; chose the best clustering with a cluster quality measure
- Cluster quality measures for metric case: (countless variants exist in literature; for an overview: e.g. [2]) (Objective functions modeling clustering paradigm):
  - Dunn-Index
  - Entropy based indices
  - ....

**Dunn Index:**

\[
D = \min_{k_1 \in [1,K]} \left( \min_{k_2 \in [1,K]} \left( \frac{d_1(C_{k_1}, C_{k_2})}{\max_{k_3 \in [1,K]} d_2(C_{k_3})} \right) \right)
\]

where \(d_1(C_{k_1}, C_{k_2})\) is the distance function between two clusters defined by

\[
d_1(C_{k_1}, C_{k_2}) = \min_{i \leq j, (n_1, n_2) \in C_{k_1} \land x_{n_1} \in C_{k_2}} \| x_{n_1} - x_{n_2} \|
\]

(that is the single link distance from SAHN).
The “diameter” \(d_2\) of the clusters is defined by

\[
d_2(C_i) = \max_{(n_1, n_2) \in C_i} \| x_{n_1} - x_{n_2} \|
\]
Dunn Index:

\[ D = \min_{k_1 \in [1,K]} \left( \min_{k_2 \in [1,K]} \left( \frac{d_1(C_{k_1}, C_{k_2})}{\max_{k_3 \in [1,K]} d_2(C_{k_3})} \right) \right) \]

where \( d_1(C_{k_1}, C_{k_2}) \) is the distance function between two clusters defined by

\[ d_1(C_{k_1}, C_{k_2}) = \min_{\{(n_1,n_2)|x_{n_1} \in C_{k_1}, x_{n_2} \in C_{k_2}\}} \|x_{n_1} - x_{n_2}\| \]

(that is the single link distance from SAHN).

The “diameter” \( d_2 \) of the clusters is defined by

\[ d_2(C_k) = \max_{\{(n_1,n_2)|x_{n_1} \in C_k, x_{n_2} \in C_k\}} \|x_{n_1} - x_{n_2}\| \]
K-Means is "OK" as cluster algorithm, but has certain disadvantages:
- favors spherical clusters
- need to know K
- no notion of noise

Alternative → DBSCAN [4]
(de facto state of the art):
- Idea: Two parameters: minPt, \( \epsilon \)
- Rough idea: iterate:
  visit previously unseen pattern \( x \):
  - if in \( \epsilon \)-neighborhood \( \{x'\} \) of \( x \): \(|\{x'\}| \geq \text{minPt}\) then
    start new cluster: include \( x \) and \( \{x'\} \) and those of their
    \( \epsilon \)-neighborhoods \( \{x''\} \) that are dense enough (\(|\{x''\}| \geq \text{minPt}\)), etc.
  - else: \( x \) is noise

K-Means is "OK" as cluster algorithm, but has certain disadvantages:
- favors spherical clusters
- need to know K
- no notion of noise

Alternative → DBSCAN [4]
(de facto state of the art):
- Idea: Two parameters: minPt, \( \epsilon \)
- Rough idea: iterate:
  visit previously unseen pattern \( x \):
  - if in \( \epsilon \)-neighborhood \( \{x'\} \) of \( x \): \(|\{x'\}| \geq \text{minPt}\) then
    start new cluster: include \( x \) and \( \{x'\} \) and those of their
    \( \epsilon \)-neighborhoods \( \{x''\} \) that are dense enough (\(|\{x''\}| \geq \text{minPt}\)), etc.
  - else: \( x \) is noise
**K-Means Clustering**

- K-Means is "OK" as cluster algorithm, but has certain disadvantages:
  - favors spherical clusters
  - need to know K
  - no notion of noise

- Alternative → DBSCAN [4] (de facto state of the art):
  - Idea: Two parameters: minPt, ε
  - Rough idea: iterate:
    - visit previously unseen pattern x:
      - if in ε-neighborhood (x') of x: ||x'|| ≥ minPt then
        - start new cluster: include x and (x') and those of their ε-neighborhoods (x'') that are dense enough (||x''|| ≥ minPt), etc.
      - else: x is noise

**Fuzzy C-Means Clustering**

- K-Means was a crisp algorithm. Now: fuzzy variant
- Reformulate K-Means objective function with membership matrix
  - $r_{nk}$: Membership of pattern $x_n$ in class $C_k$
  - $J_{SQE} = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||x_n - \mu_k||^2$

- optimization criterion
  - $\frac{dJ_{SQE}}{d\mu_k} = 0$
  - together with non-overlapping constraint
    - $\forall n(\exists k(r_{nk} = 1) \land ((k' \neq k) \rightarrow (r_{nk'} = 0)))$

- leads to well known K-Means
  - $\mu_k = \frac{\sum_{n=1}^{N} r_{nk} x_n}{\sum_{n=1}^{N} r_{nk}/|C_k|} \sum_{n| x_n \in C_k} x_n$

**K-Means Clustering**

- Example Application: Clustering locations
  - Problem: How do we distinguish socially relevant clusters (candidates for groups) from quasi groups?
    - Compute clusterings over period of time: Good candidates: clusters that appear over and over again, clusters that appear periodically
    - Establish threshold for distance in clusters: Human "social distance": A few meters (if groups are very small); few tens of meters (if groups are medium sized)
    - Include velocities: If divergent → no group
Fuzzy C-Means Clustering

Now modify objective function to:

\[ J_{GSQE} = \sum_{n=1}^{N} \sum_{k=1}^{K} (r_{nk})^m \|x_n - \mu_k\|^2 \]

Exponent \( m \) models degree of fuzzyness:
- \( m \to 1 \): K-Means (crisp case);
- \( m \to \infty \): \( r_{nk} \to 1/K \) (where K is the number of clusters)

Optimize the obj. fct. under the conditions:

\[ \forall x_n : \sum_{k=1}^{K} \alpha_k(x_n) = \sum_{k=1}^{K} r_{nk} = 1 \]
\[ \forall C_k : \sum_{n=1}^{N} \alpha_k(x_n) = \sum_{n=1}^{N} r_{nk} > 0 \]

Result:

\[ r_{nk} = \left( \sum_{k' = 1}^{K} \left( \frac{\|x_n - \mu_{k'}\|^2}{\|x_n - \mu_{k'}\|^{2m-1}} \right)^{-\frac{m}{m-1}} \right)^{\frac{1}{m-1}} \]  
\[ \mu_k = \frac{\sum_{n=1}^{N} r_{nk} x_n}{\sum_{n=1}^{N} r_{nk}} \]

the result assumes that no patterns and prototypes coincide

\[ \forall n, k : \|x_n - \mu_k\| \neq 0 \]

if they do coincide, set \( r_{nk} = 1 \) for \( x_n = \mu_k \) and \( r_{nk} = 0 \) for \( x_n \neq \mu_k \)
Fuzzy C-Means Clustering

• Limit \( m \to \infty \) gives:
  \[
  r_{nk} \xrightarrow{m \to \infty} \frac{1}{\sum_{k'=1}^{K} 1} = \frac{1}{K}
  \]

• Limit \( m \to 1 \) we get the nearest neighbor rule (K-Means) because:
  \[
  r_{nk} = \frac{1}{(\sum_{k' \neq k} \left( \frac{\|x_n - \mu_k\|^2}{\|x_n - \mu_{k'}\|^2} \right)^{\frac{2}{m-1}} + 1) ^\frac{1}{m-1}}
  \]
  in the limit \( m \to 1 \) the first sum in the denominator becomes \( \infty \) if
  \[
  \|x_n - \mu_k\| \neq \min_{1 \leq k' \leq K} \|x_n - \mu_{k'}\|
  \]
  and it becomes 0 if
  \[
  \|x_n - \mu_k\| = \min_{1 \leq k' \leq K} \|x_n - \mu_{k'}\|
  \]
Gaussian Mixture Models

- Fuzzy C-Means is “OK” as a non-crisp clustering alg, but (as K-Means) favors spherical clusters $\rightarrow$ better approaches

**Example:** Gaussian Mixture Models (GMM)

- Linear combination of Gaussians
  \[
p(x) = \sum_{k=1}^{K} \pi_k N(x | \mu_k, \Sigma_k), \quad \text{where} \quad \sum_{k=1}^{K} \pi_k = 1, \quad 0 \leq \pi_k \leq 1
  \]

![Diagram of Gaussian mixture models](image)

![Diagram of Gaussian mixture models](image)

Gaussian Mixture Models

- Fuzzy C-Means is “OK” as a non-crisp clustering alg, but (as K-Means) favors spherical clusters $\rightarrow$ better approaches

**Example:** Gaussian Mixture Models (GMM)

- Linear combination of Gaussians
  \[
p(x) = \sum_{k=1}^{K} \pi_k N(x | \mu_k, \Sigma_k), \quad \text{where} \quad \sum_{k=1}^{K} \pi_k = 1, \quad 0 \leq \pi_k \leq 1
  \]

![Diagram of Gaussian mixture models](image)

![Diagram of Gaussian mixture models](image)

GMM-Basics

- Responsibilities
  \[
  \gamma(z_k) \equiv p(z_k = 1 | x) = \frac{p(z_k = 1)p(x | z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(x | z_j = 1)} = \frac{\pi_k N(x | \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x | \mu_j, \Sigma_j)}
  \]

**Example**

![Example plots](image)

![Example plots](image)
**Example:** $x \in \mathbb{R}^n$ and $p(x|\theta)$ is one multivariate Gaussian

$$p(x|\theta) = \mathcal{N}(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{ \frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\}$$

- **log likelihood:** (on base e)

$$\ln p(X|\theta) = \ln p(X|\mu, \Sigma) = - \frac{ND}{2} \ln(2\pi) - \frac{N}{2} \ln |\Sigma| - \frac{1}{2} \sum_{n=1}^{N} (x_n - \mu)^T \Sigma^{-1} (x_n - \mu)$$

- **Maximum log likelihood:**

$$\Theta_{ML} = \arg\max_{\Theta} \log p(X|\theta) \Rightarrow \nabla_{\theta} \left( \sum_i \log p(x_i|\theta) \right) = 0$$

$$\mu_{ML} : \frac{\partial}{\partial \mu} \ln p(X|\mu, \Sigma) = 0 \Rightarrow \mu_{ML} = \frac{1}{N} \sum_{n=1}^{N} x_n$$

$$\Sigma_{ML} : \frac{\partial}{\partial \Sigma} \ln p(X|\mu, \Sigma) = 0 \Rightarrow \Sigma_{ML} = \frac{1}{N} \sum_{n=1}^{N} (x_n - \mu_{ML})(x_n - \mu_{ML})^T$$