14: Clique Finding
Machine Learning and Real-world Data (MLRD)

Ryan Cotterell
(based on slides created by Simone Teufel)

Lent 2020
Last session: betweenness centrality

- You implemented betweenness centrality.
- This let you find “gatekeeper” nodes in the Facebook network.
- We will now turn to the task of finding clusters in networks.
- You will test this on a small network derived from one Facebook user.
Clustering in networks

- **clustering**: automatically grouping data according to some notion of closeness or similarity.

- **agglomerative clustering** works bottom-up.

- **divisive clustering** works top-down, by splitting.

- Newman-Girvan method — a form of divisive clustering.

- Criterion for breaking links is edge betweenness centrality.

- When to stop?
  - Prespecified (today’s tick): use prior knowledge to decide when to stop, based on number of clusters.
  - Inherent ‘goodness of clustering’ metric: today’s starred tick uses **modularity** (Newman 2004).
Step 1: Code for determining connected components

- Today’s graph is disconnected: there are five connected components.
- Finding connected components: depth-first search, start at an arbitrary node and mark the other nodes you reach.
- Repeat with unvisited nodes, until all are visited.
- Implementation hint: depth-first, so use recursion (the program stack stores the search state).
Step 2: Edge betweenness centrality

- Previously: $\sigma(s, t | v)$ — the number of shortest paths between $s$ and $t$ going through node $v$.
- Now: $\sigma(s, t | e)$ — the number of shortest paths between $s$ and $t$ going through edge $e$.
- Algorithm only changes in the bottom-up (accumulation) phase: $\delta(v)$ much as before, but $c_B[(v, w)]$
Brandes (2008) pseudocode

\begin{algorithm}
\caption{Accumulation \textit{//} back-propagation of dependencies}
\begin{algorithmic}
\For{$v \in V$} \State $\delta[v] \leftarrow 0$
\While{$S$ not empty} \State pop \, $w \leftarrow S$
\For{$v \in \text{Pred}[w]$} \State $\delta[v] \leftarrow \delta[v] + \frac{\sigma[v]}{\sigma[w]} \cdot (1 + \delta[w])$
\EndFor
\If{$w \neq s$} \State $c_B[w] \leftarrow c_B[w] + \delta[w]$
\EndIf
\EndWhile
\EndFor
\end{algorithmic}
\end{algorithm}

Edge betweenness

\textbf{output:} betweenness $c_B[q]$ for $q \in V \cup E$ (initialized to 0)

\begin{algorithm}
\caption{Accumulation}
\begin{algorithmic}
\For{$v \in V$} \State $\delta[v] \leftarrow 0$
\While{$S$ not empty} \State pop \, $w \leftarrow S$
\For{$v \in \text{Pred}[w]$} \State $c \leftarrow \frac{\sigma[v]}{\sigma[w]} \cdot (1 + \delta[w])$
\State $c_B[(v, w)] \leftarrow c_B[(v, w)] + c$
\State $\delta[v] \leftarrow \delta[v] + c$
\EndFor
\If{$w \neq s$} \State $c_B[w] \leftarrow c_B[w] + \delta[w]$
\EndIf
\EndWhile
\EndFor
\end{algorithmic}
\end{algorithm}

ignore last line
Step 3: Newman-Girvan method

while number of connected subgraphs < specified number of clusters (and there are still edges):

1. calculate edge betweenness for every edge in the graph
2. remove edge(s) with highest betweenness
3. recalculate number of connected components

Note:

- Treatment of tied edges: either remove all (today) or choose one randomly.
Visualization as dendrogram

- Either: stop at prespecified level (tick).
- Or: complete process and choose best level by ‘modularity’ (starred tick).

Dolphin data: different clustering layers

- squares vs circles: first split
- different colours: further splits

Facebook circles dataset: McAuley and Leskovec (2012)

- Designed to allow experimentation with automatic discovery of circles: Facebook friends in a particular social group.
- Profile and network data from 10 Facebook ego-networks (networks emanating from one person: referred to as an ego).
- Gold-standard circles, manually identified by the egos themselves.
- Average: 19 circles per ego, each circle with average of 22 alters.
- Complete network consists of 4,039 nodes in 193 circles.
Facebook circles

Requires more sophisticated methods than Newman-Girvan: a) nodes may be in multiple circles, b) not just network data.

25% of circles are contained completely within another circle
50% overlap with another circle
25% have no members in common with any other circle
Evaluating simple clustering

- Assume data sets with gold standard or ground truth clusters.
- But: unlike classification, we don’t have labels for clusters, number of clusters found may not equal true classes.
- **purity**: assign label corresponding to majority class found in each cluster, then count correct assignments, divide by total elements (cf accuracy).


- But best evaluation (if possible) is **extrinsic**: use the system to do a task and evaluate that.
Clustering and classification

- Classification (e.g., sentiment classification): assigning data items to predefined classes.
- Clustering: groupings can emerge from data, unsupervised.
- Clustering for documents, images etc: anything where there’s a notion of similarity between items.
- Most famous technique for hard clustering is k-means: very general (also variant for graphs).
- Also soft clustering: clusters have graded membership
Task 12:
- Implement the Newman-Girvan method.
- Discover clusters in the network provided.