GoWvis: a Web App for Graph-based Text Visualization & Summarization

https://safetyapp.shinyapps.io/GoWvis/
Antoine J.-P. Tixier, Konstantinos Skianis, Michalis Vazirgiannis
Computer Science Laboratory, École Polytechnique, France

---

**Introduction**

Graph-of-Words (GoW) fundamentals:
- statistical approach based on the Distributional Hypothesis
- edge between two terms if they co-occur within a fixed-size sliding window
- encodes term dependence strength (via edge weights) and term order (via edge direction)
- enables graph theory to be applied to text
- linear in time and space (resp. \(O(nW), O(n + m)\))

GoW proved highly successful:
- keyword extraction and summarization [Mihalcea & Tarau 2004, Rousseau & Vazirgiannis 2015]
- information retrieval [Rousseau & Vazirgiannis 2013]
- document classification [Malliaros & Skianis 2015, Rousseau et al. 2015]
- and more...

Motivation for GoWvis:
- GoW can be used to improve almost any NLP task...
- ... but it has many pre-processing, graph building, and graph mining parameters

...there are needs to interactively explore the parameter space

---

**I. Text pre-processing**

- Keep only nouns and adjectives? Boolean, defaults to TRUE
- Remove SMART stopwords? Boolean, defaults to TRUE
- Stemming? Boolean, defaults to TRUE. If used, tends to yield smaller and denser graphs.

The surviving terms are used as the nodes of the graph-of-words

---

**II. Graph building**

- Window size. Integer between 2 and 12, defaults to 3. The larger the window, the denser the graph.
- Build on processed text? Boolean, defaults to TRUE. If used, tends to link more distant words and produce denser graphs.
- Overspan sentences? Boolean, defaults to TRUE. If FALSE, two words can only co-occur if they belong to the same sentence.

---

**III. Graph mining: community detection**

Goal: cluster the graph-of-words into groups within which connections are dense and between which they are sparse

The clusters match the topics and sub-topics within the document

In practice: retaining only the main communities improves coverage and removes noise

- Algorithm? List, defaults to “none”. Choices are “fast greedy”, “louvain”, “walktrap”, “infomap”, “label prop” and “none”
- Size threshold? Numeric (from 0.4 to 1.0, by 0.1), defaults to 0.8. Percentile size threshold used to determine which communities should be considered as main ones.
- Weighted? Boolean, defaults to FALSE. Whether edge weights should be used.
- Directed? Boolean, defaults to FALSE. Whether edge direction should be used (only available for “infomap”).

---

**IV. Graph mining: degeneracy**

**K-CORE DECOMPOSITION**
- a k-core of \(G = (V,E)\) is a maximal connected subgraph of \(G\) in which every vertex \(v\) has at least degree \(k\) [Seldman 1963]
- \(v\) has core number \(k\) if it belongs to the \(k\)-core but not to the \((k+1)\)-core
- the \(k\)-core decomposition of \(G\) is the set of all its cores from 0 (\(G\) itself) to \(k_{\text{max}}\) (its main core)
- complexity: \(O(n + m)\) resp. \(O(m \log(n))\) in time in the (un)weighted cases, \(O(n)\) in space [Batagelj & Zaversnik 2002]

**K-TRUSS DECOMPOSITION**
- a \(k\)-truss of \(G = (V,E)\) is the largest subgraph of \(G\) in which every edge \(e\) belongs to at least \(k - 2\) triangles [Cohen 2008]
- \(e\) has truss number \(k\) if it belongs to the \(k\)-truss but not to the \((k+1)\)-truss
- the truss number of \(v\) is the maximum truss number of its adjacent edges
- the \(k\)-truss decomposition of \(G\) is the set of all its \(k\)-trusses from \(k - 2\) to \(k_{\text{max}}\)
- complexity: \(O(m^{1+\epsilon})\) in time and \(O(m + n)\) in space [Wang & Cheng 2012]

---

[Image of graph communities]