Evidence for widespread thematic structure in the mental lexicon: supplemental information

Simon De Deyne[a], Steven Verheyen[b], Amy Perfors[a] and Daniel J. Navarro[a]
[a] University of Adelaide, School of Psychology, 5005 Adelaide, Australia
[b] University of Leuven, Department of Psychology, Tiensestraat 102, 3000 Leuven, Belgium

Tuesday 3rd February, 2015

Background

Many real networks exhibit parts that are more tightly connected than others. The goal of network clustering is to identify clusters with members that are more strongly inter-connected than connected to the rest of the network. The resulting solution should thus exhibit denser interconnectivy within clusters than between clusters. Network clustering is a promising technique that only recently has allowed us to investigate large-scale directed and weighted graphs. At the same time it is notoriously complex and a single metric to determine the structure in a graph is unlikely to be applicable to any kind of graph (see the extensive review of the state-of-the-art in Fortunato, 2010). However, at the moment of writing the *Order Statistics Local Optimization Method* (OSLOM) leads to superior results on directed graphs and the detection of strongly overlapping clusters, which makes it particularly suited for the study of graphs derived from word association data. Below we provide an informal summary of how this algorithm works, but refer to the original article (Lancichinetti, Radicchi, Ramasco, & Fortunato, 2011) for full details.

The key feature of OSLOM in comparison to other approaches is that it uses the significance of a cluster to evaluate the fitness of this cluster. This is operationalized by the probability of finding the cluster in a random null model that does not exhibit a clustered structure. The random model employed by OSLOM is the directed weighted extension of the configuration model (Radicchi, Lancichinetti, & Ramasco, 2010). In contrast to some other approaches, the algorithm works locally, meaning that it starts by evaluating a single vertex randomly selected from the graph. Next q vertices are added to it, where q is taken from a power law distribution, and the resulting cluster is then evaluated. This procedure is repeated many times, resulting in a final set of clusters that may overlap. The algorithm stops when it keeps finding similar clusters over and over. In a next step, cluster merges are determined by evaluating whether the subgraphs composed of the candidate merged clusters are considered distinct compared to the null-model or not.

Any iteration of the algorithm involves three main loops:

1. Analyze a single cluster and iteratively add or remove nodes to/from this cluster depending on a fitness score (outlined above) calculated for all nodes.

2. Check which clusters can be grouped together and determine a set of stochastic covers (i.e., overlapping clusters).

3. Derive a higher hierarchical solution using a new network obtained through previous steps.

Full technical details of the OSLOM algorithm are available in Lancichinetti et al. (2011) and Radicchi et al. (2010). The software for implementing OSLOM can be freely downloaded at `http://www.oslom.org/`.

Application

Initial clustering solutions

Initial clustering solution. One of the benefits of OSLOM is that clustering solutions found by other algorithms can be evaluated together with the clusters found by OSLOM. When clusters obtained through other algorithms are available, we can evaluate their fitness as the likelihood of such structure to arise in the appropriate null model similar to the way OSLOM evaluates its clusters. In addition, using additional solutions obtained through other techniques often speeds up the computation time.

In the accompanying article we followed this practice and clustering was performed by using solutions derived from five iterations of Infomap (Rosvall & Bergstrom, 2008), two iterations of COPRA (Gregory, 2010) and one iteration of the Louvain algorithm (Blondel, Guillaume, Lambiotte, & Lefebvre, 2008). The choice of these algorithms was based on the fact that they operate on large graphs and implementations of them are freely available and bundled with the original OSLOM software.

Parameters

Two parameters govern the results of OSLOM: p and cp. About these parameters the authors write (see Lancichinetti et al., 2011, page 4):

> The influence of the parameter values is however relevant only when the community structure of the network is not pronounced. When clusters are well defined, the results of OSLOM do not depend on the particular choice of the parameter values.

Coverage Parameter. The coverage parameter cp determines whether clusters are to be joined together or not. It was left unchanged at the default value of 0.50.

p-threshold. The significance threshold p was set at 0.25 for the reported solution, which is higher than the standard 0.1 used throughout the evaluation in Lancichinetti et al. (2011). As indicated by Lancichinetti et al. (2011), increasing the value of p results in more clusters as smaller clusters are more easily considered significant. In practice we found that varying p does lead to slightly different results at the highest hierarchical levels. Here the value for p was piloted by seeing if the clusters with the largest p-values had a clear interpretation. This follows the suggestion in Lancichinetti et al. (2011) on page 14:

> The actual resolution of the method is thus not due to the null model, but to the choice of the threshold p. In this paper we have set $p = 0.1$, which is often used in various contexts and delivers excellent performance on the benchmark graphs we have adopted. Nevertheless how much a real graph *deviated* from a random graph depends on the specific system at hand, and it would be more appropriate to estimate the threshold p case by case.

Clearly, a more principle approach to study the effect of different settings for p is something for future consideration.

Computation

Similar to other community detection techniques with a stochastic character, more accurate results are achieved through many iterations and are mostly determined by the size of the graph and the time available for computations. The number of iterations at the lowest level was set to 100 (default value was 10), whereas the number of runs to determine higher hierarchical levels was kept at the default (50). This resulted in a computation time around 3 to 4 hours on a 32GB Ram Intel® Core™ i7-2600K 3.40GHz Linux workstation.

References

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*(10), P10008+.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, *486*(3), 75–174.

Gregory, S. (2010). Finding overlapping communities in networks by label propagation. *New Journal of Physics*, *12*(10), 103018.

Lancichinetti, A., Radicchi, F., Ramasco, J. J., & Fortunato, S. (2011). Finding statistically significant communities in networks. *PloS one*, *6*(4), e18961.

Radicchi, F., Lancichinetti, A., & Ramasco, J. J. (2010). Combinatorial approach to modularity. *Physical Review E*, *82*(2), 026102.

Rosvall, M., & Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, *105*(4), 1118–1123.