CS-E5740 Complex Networks,
Answers to exercise set 8

Adam Ilyas 725819

January 3, 2019

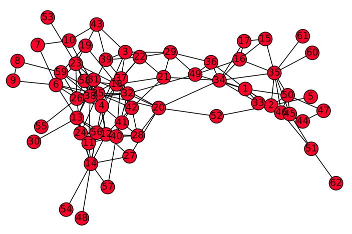
Acknowledgements: Zeyneb Erdogan

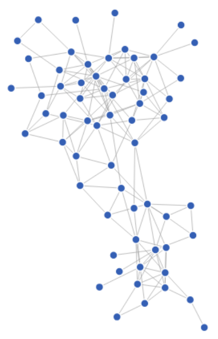
1 Community detection overview

In this exercise, we will get some hands-on experience on community detection. You will be studying two networks, the network of dolphins `dolphins.edg` and an artificial graph `lfr100.edg` with a built-in community structure `lfr100.cmtys`.

Part 1: Detecting communities in the dolphin network

a) Upload first the dolphin network to Jako, and **visualize** the bare network (In Jako CD, you can visualize networks by applying the `NullCD` community on the network.)





Comment the visualization based on your personal impression: are there any well defined communities in this network? How would you partition the network into different communities using only your own intuition?

Ans. We can observe 2 communities, one of the the community is larger and has more connections than the other. The 2 communities are connected by just a few nodes (dolphins) and these few connections are easy to spot.

Lets next find out whether some algorithms are able to discover similar communities as you identified. Do this by applying four different community detection methods on the graph:

b) Use the built-in visualization tool in `Jako CD` to visualize the community structures obtained using the above methods.

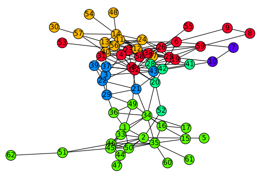


Figure 1: Infomap (Edler and Rosvall)

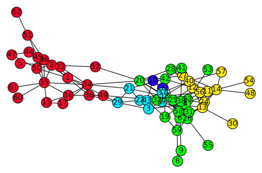


Figure 2: Girvan-Newman

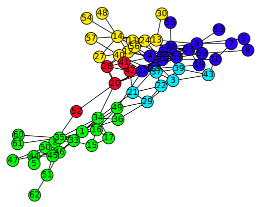


Figure 3: Louvain (original code)

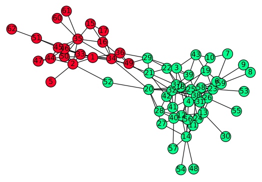


Figure 4: Stochastic Block Model

c) Do the different methods give out same results? Do the results correspond to your intuition of what the modules should be as you described before?

Ans.

- Infomap, Girvan-Newman and lovain gives similar models, detecting 5 to 6 communities.
- Stochastic block gives 2 communities and how I would intuitively partitiion the 2 communities

Part 2: Comparing communities and benchmarking

d) Apply the first three algorithms of Part 1 to the LFR network in Jako CD and **visualize** the community structures. **Comment** on how easy it is to assess the performance of the algorithms visually.

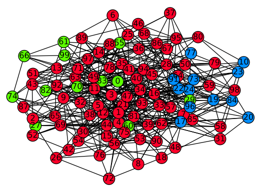


Figure 5: Infomap (Edler and Rosvall)

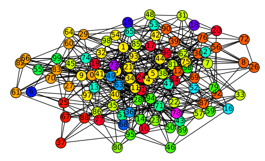


Figure 6: Girvan-Newman

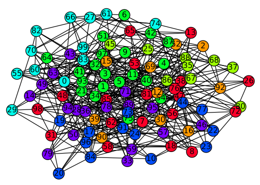


Figure 7: Louvain (original code)

Ans. It is difficult to assess the performance of the algorithms because the network is dense and visually, it looks messy. Hence, it is hard to observe degree and which are the neighbours of each nodes.

e) To get familiar with VI variation of information (VI), lets first consider two partitions of 12 nodes:
A = {1: [1,2,3,4,5,6,7,8,9], 2: [10,11,12]}
B = {1: [1,2,3,4,5,6,7], 2: [8,9,10], 3: [11,12]}.

Calculate VI between these partitions manually (pen and paper).

$$\begin{aligned}
H(A) &= -\frac{9}{12}\log_2\frac{9}{12} - \frac{3}{12}\log_2\frac{3}{12} = 0.811 \\
H(B) &= -\frac{7}{12}\log_2\frac{7}{12} - \frac{3}{12}\log_2\frac{3}{12} - \frac{2}{12}\log_2\frac{2}{12} = 1.345 \\
I(A,B) &= \frac{7}{12}\log_2\frac{84}{63} + \frac{2}{12}\log_2\frac{24}{27} + \frac{1}{12}\log_2\frac{12}{9} + \frac{2}{12}\log_2\frac{24}{6} = 0.439 \\
VI(A,B) &= H(A) + H(B) - 2I(A,B) = 0.811 + 1.345 - 0.879 \\
&= 1.361
\end{aligned}$$

f) Next, write a Python function for calculating VI between two partitions. Use the functions for calculating entropies and mutual information in the code template (`com_det_benchmarking.py`) available in MyCourses. Before applying your code on the actual community structures we are interested in, test your function by making sure that your code for the VI produces the following results:

i) VI between same partitions equals to zero

ii) VI between a partition in which all nodes belong to the same and a partition in which all nodes belong to different communities equals $log_2 N$.

iii) VI between the partitions in e) equals the result you calculated manually.

```
def vi(cmtys1, cmtys2):
    VI=entropy(cmtys1)+entropy(cmtys2)
    -2*mutual_information(cmtys1,cmtys2)
    return VI

# First step: testing the code:
test1 = {1:[1, 2, 3, 4, 5, 6, 7, 8, 9], 2:[10, 11, 12]}
test2 = {1:[1, 2, 3, 4, 5, 6, 7], 2:[8, 9, 10], 3:[11, 12]}

print(vi(test1, test2))
print(vi(test1, test1))
print(vi(test2, test2))

Out[]:
1.03230129690871
2.220446049250313e-16
0.0
```

g) **Compute** the pairwise distances between all the network partitions (Infomap, Louvain, Girvan-Newman, Stochastic block model, ground-truth). Comment on the results you have obtained. Which method performs best in terms of the VI measure?

```
Ground vs Infomap: 0.923
Ground vs Girvan-Newman: 2.144
Ground vs Louvain: 1.516
Ground vs Stochastic block model: 2.763
Infomap vs Ground: 0.923
Infomap vs Girvan-Newman: 1.988
Infomap vs Louvain: 1.024
Infomap vs Stochastic block model: 2.768
Girvan-Newman vs Ground: 2.144
Girvan-Newman vs Infomap: 1.988
Girvan-Newman vs Louvain: 2.468
Girvan-Newman vs Stochastic block model: 3.684
Louvain vs Ground: 1.516
Louvain vs Infomap: 1.024
Louvain vs Girvan-Newman: 2.468
Louvain vs Stochastic block model: 2.765
Stochastic block model vs Ground: 2.763
Stochastic block model vs Infomap: 2.768
Stochastic block model vs Girvan-Newman: 3.684
Stochastic block model vs Louvain: 2.765
```

Model output from Infomap algorithm is the best performing since the VI(infomap, ground) has the value closest to 0 as compared to the other models.

2 Modularity

a) **Calculate** the value of modularity for the two partitions shown in Fig. 2 (in the first, there are two clusters, and in the second, the whole network is taken as a single cluster).

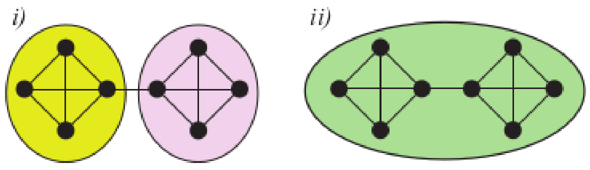


Figure 2: Two module configurations for a)

i) $Q = \frac{6}{13} - (\frac{13}{13\times 2})^2 = 0.423$

ii) $Q = 1 - (\frac{26}{13\times 2})^2 = 0$

b) Write ΔQ as a function of $L, d_a, d_b, d_{ab}, l_a, l_b,$ and l_{ab}.

$$\Delta Q = \frac{l_{ab}}{L} - (\frac{d_{ab}}{2L})^2 - \frac{l_a}{L} + \frac{d_a}{2L}^2 - \frac{l_b}{L} + (\frac{d_b}{2L})^2$$