Annotated Bibliography

Ian Chen

Contents

1 Community Detection

Well-Connectedness and Community Detection

Park et al. [0] is addressing the problem that standard community detection algorithms are returning *poorly-connected* clusters. They perform experiments on real-world and simulated networks for a suite of different methods. Here, I will focus on the Leiden algorithm (optimizing Constant-Potts (CPM), or Modularity (Mod))..

To improve the connectivity of output clusters, they introduce a new pipeline, Connectivity Modifier (CM). CM applies post-processing (according to user inputted parameters) that can work with a variety of base methods. While there are poorly-connected clusters, CM will uncluster them and recluster individually.

First, they explore some properties of the Leiden algorithm [0]. They found that Leiden+Mod behaved like Leiden+CPM with very small resolution (<1e-5), in terms of connectivity and sizes of the clusters. In general, refining the resolution (making it smaller) leads to more coverage (and larger size) of clusters, at the expense of their connectivity.

To formally define well-connectedness, they chose a threshold function $f(n) = log_2(n)$, where n is the cluster size, to decide whether a cluster was well-connected or not. With this, indeed Leiden only produces mostly well-connected clusters with resolution $r \geq 0.1$, on the real-world datasets.

On real-world networks, applying CM decreases the node coverage to varying degrees based on the dataset and method. For coarse resolutions, CM does not significantly affect the results (indeed, there is a theoretical guarantee on connectedness). For fine resolutions, CM significantly drops the coverage, as the frequency of poorly-connected clusters increases. CM primarily *reduces* the cluster size, but ocassionally it *degrades* the cluster into singletons, or *splits* into multiple smaller clusters.

The stages within the CM pipeline affect the output accuracy in different ways. First, the initial pre-filter of clusters with ≤ 1 edge connectivity decreases the accuracy. However, the resulting ability to improve poorly-connected clusters is neutral of beneficial to accuracy. Hence, for fine resolutions, we see improvements in accuracy (because higher proportion of clusters are poorly-connected), and minor drops in accuracy for coarse resolutions.

Overall, their results provide further evidence of *over-clustering* and the accuracy downfalls. Their CM method showcases the potential of post-processing as a way to mitigate it.

CM++ - A Meta-method for Well-Connected Community Detection

Ramavarapu [0] describes the codebase for CM [0]. The CM pipeline hapens in two stages:

- Run VieCut [0] to find a small edge cut.
- If the edge cut is below some threshhold $f(n)$, remove the cut and recluster the partitions
- Repeat...

The codebase is implemented in Python, with support for multiple cores.

Improved Community Detection Using Stochastic Block Models

Park et al. [0] builds upon the previous work in the CM pipeline [0]. Indeed, they examine three treatments here:

- CC (Connected Components), each of the connected components represent an individual cluster
- WCC (Well-Connected Clusters), removing minimum cuts until well-connected
- CM (Connectivity-Modifier), WCC but with an additional re-clustering after removing a mincut

They first explore the properties of SBM clusters, and then evaluate their treatments on both real and synthetic networks. Their findings are different for small vs medium/large networks.

On small (≤ 1000 nodes) networks, SBM typically returns clusterings that are connected, and sometimes well-connected. Moreover, there are only small drops in node coverage due to the treatments (with CC affecting the least).

On the medium and large networks, SBM typically returns clusters that are disconnected. Here, the treatments do greatly drop the node coverage (indeed the magnitude is lowest for CC, then WCC, then CM). Note that the average returned cluster size after CM is larger than that for CC and WCC (cause it does re-clustering). Indeed, CC drops the average cluster size by an order of magnitude (or more).

To determine the effect of the treatments on accuracy, they further evaluate on synthetic (LFR) networks. Towards ARI, CC typically has neutral impact, WCC has neutral or beneficial, and CM's impact is unpredictable.

For theoretical explanation, they look at the description length formula. They sum the negative log-likelihood of the model (adjacency matrix given parameters) and of the priors. Specifically, for the edge-count matrix,

$$-\log p(e) = \log \binom{B(B+1)/2 + E - 1}{E}$$

where B and E are the number of blocks and edges. Hence, clusterings before the treatment are favored by the description length formula (despite the post-treatment having higher accuracy).

Ensemble clustering for graphs: comparisons and applications

Poulin and Theéberge [0] further evaluate their previously published ensemble clustering method ECG [0].

They address some well known issues of modularity, including

1. The resolution limit — it tends to underfit when there are lots of clusters
2. Stability between iterations of Louvain

They do this by defining a weighting scheme: given multiple partitions, they form weights through co-association,

$$w_{\mathcal{P}}(u,v) = \begin{cases} w_* + (1-w_*)\left(\frac{1}{k}\sum_{i=1}^{k} \mathbf{1}_{(u,v)\in P_{\mathcal{P}}}\right) & \textbf{otherwise } (u,v) \in 2-\text{core of } \mathsf{G} \\ w_* & \textbf{otherwise} \end{cases}$$

i.e. counting how many partitions place u and v together. On this weighted graph, they then run the Louvain algorithm to obtain a final clustering.

They show very positive results on all grounds, including on a (dubious) evaluation on real networks. ECG is better especially on low signal (e.g. LFR graphs with high mixing). Moreover, they show applications of their weighting beyond obtaining a partition. For example, the kurtosis of the weighting distribution is a good indicator for community structure within a network. In addition, this can be used for community search by using the weights to focus on specific seeds.

There are other issues of modularity, such as connectivity and homogeneity of clusters. Does ECG improve on these at all?

Recompile

Defining and evaluating network communities based on ground-truth

Yang and Lescovec [0] describe their metholodology in processing the ground truths in the SNAP database [0].

Recall that for community detection algorithms, we often use synthetic networks with ground truth partitions. For example, in the LFR software [0], the ground truth nodes have different connection probabilities (governed by the mixing parameter).

In empirical networks, we can use metadata to find a gold-standard labeling. For social networks (e.g. Youtube), users are nodes, friendships are edges, and user-created groups are communities. Despite this, we do not know if there are detectable topological differences between the gold-standard labeling (often by metadata).

The authors study scoring metrics for how "community-like" their clustering is. For individual clusters, they calculate the density, separability, and connectivity. They found that for their networks,

1. There is a big drop in quality after the top 5000 community (average rank across their metrics)
2. Some networks have communities that are highly separable, while others are highly dense (sometimes not both)
3. Modularity is inversely proportional to density, separability, and connectivity functions
4. Triangle Participation Rate (cohesiveness) and Conductance (separability) are the two metrics to use
    (a) They measure different qualities of communities
    (b) They are the least sensitive

Overall, this paper definitely feels a bit ad-hoc and the details feel like they're lacking. I'm not sure why they make a distinction between cluster scoring and cluster goodness. This top 5000 community threshold seems arbitrary and not something generalizable to different network sizes. Why do they include the "bad" scoring functions (e.g. modularity) into their top 5000 ranking, instead of just using TPR and Conductance. Is there a better way to weight, than just taking the average?

I read this paper to determine if and how can use these networks to measure accuracy for community detection. Reading this, I'm still not sure.

On another note, Justin Luke took a class with Lescovec! The software is publicly accessible, packaged and documented [0].

2 Synthetic Network Generation

EC-SBM Synthetic Network Generator

Hello [0]

3 Scientometrics

The pivot penalty in research

[0]