Spectral Techniques for Community Detection on 2-Hop Topology

HU, Pili

March 21, 2012

Abstract

In one of our former project[3], we formulated the problem of community detection on 2-hop topology. Given one observer in a social network, and the graph it can reach within two steps, we want to determine for those vertices whether they're in the same community with the observer. The previous project extracts features like Common Neighbours, Ademic/Adar Score, Pagerank, and Personalized Pagerank, etc. Then several neural networks are trained to combine those features. Interested readers can refer to [3] for more information.

In this project, we augment our research based on the same settings. Several spectral techniques are leveraged to tackle with the 2-hop problem. For instance, spectral clustering may utilize the information hidden in more eigen vectors. Other technique like personalized PageRank can be studied with the budgeted learning context. For data, codes, and other materials, please refer to our open source repository[2].

Contents

1 Introduction

1.1 Problem

Notations:

Table 1: Notations

Symbol	Explanation
n	node number
m	edge number
A	adjacency matrix
A_{ij}	1 if node i and node j are directly connected, 0 otherwise
$N(i)$	neighbourhood of i, namely $N(i) = \{j \mid A_{ij} = 1\}$
$d(i)$	degree of node i $d(i) = \sum_j A_{ij}$
o	observer
l_i	the real label of node i
L_i	the predicted label of node i
$N2(i)$	the 2-hop neighbours we study, defined as $\cup_{j \in N(i) \cup \{i\}} N(j)$

Problem settings:

- Give observer o.
- Give the two hop topology, namely the $N2(o)$ and the associated links.
- Try to predict L_i according to the above information.
- Ground truth l_i is directly crawled from the website.

1.2 Evaluation

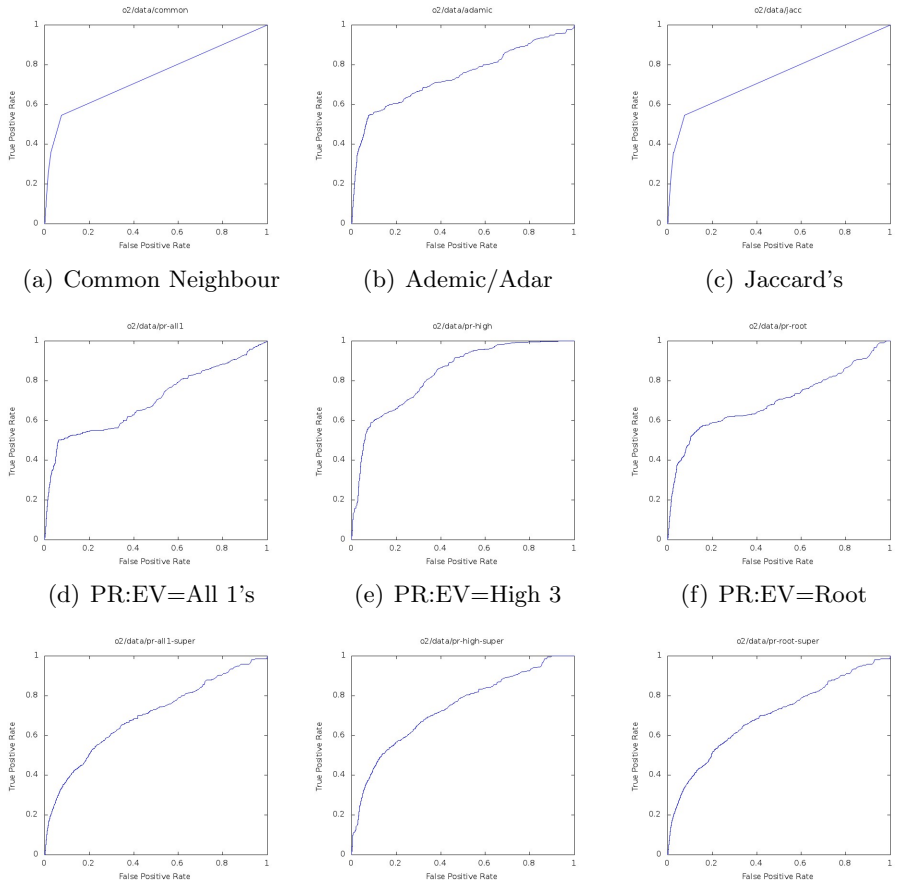
Our evaluation method distinguishes this work from many previous ones. Usual evaluation criterion is defining a certain quality functions, like[1] normalized cut, conductance, and modularity. Different research groups have their own taste and preference.

The major problem is, those quality functions can reflect the accuracy of algorithms only to a limited extent. Let spectral clustering be an example. Luxburg[6] explains clustering using the notion of finding sets of nodes with small conductance in between. In this way, if conductance is selectd as the quality function, a good "clustering" is a natural consequence of spectral clustering algorithm.

While those quality functions are simple and intuitive to facilitate theoretical study, they can not completely reflect the real application performance. In this work, we crawled ground truth data from one SNS, thus we're able to evaluate the output against those crawled label.

1.3 Digest of Previous Results

Overall speaking, we didn't get satisfying result in the previous project. The diffusion matrix after combining 9 features with neural network always show a tradeoff between precision and recall.



(a) Common Neighbour (b) Ademic/Adar (c) Jaccard's

(d) PR:EV=All 1's (e) PR:EV=High 3 (f) PR:EV=Root

(g) PR:EV=All 1's(+SN) (h) PR:EV=High 3 (+SN) (i) PR:EV=Root(+SN)

Figure 1: ROC Curve of 9 Features

Fig(1) shows the single feature evalutation result. By varying a threshold, we plot the True Positive Rate v.s. the False Positive Rate(label 1, namely "same community"). Those single feature evaluations are the useful part of the former project. The observations are:

- Three simple features, as claimed by some researchers already performs well in practice.
- Compared with those simple features, more complicated PR series exhibits little benefit.
- Among the 6 PR variations, the best performed one is Personalized PR with escape vector equal to 3 highest degree nodes from the target community.

2 Proposed Future Study Line

2.1 Evaluation Modification

From previous study, we find that diffusion matrix is not the most convenient way to do evaluation. The diffusion matrix is the final result, but highly depends on choice of parameters(may be trained weights in neural network). In this work, we want to leverage spectral techniques and do a in depth study of them. We find that ROC Area(ROCA), namely the area under ROC curve, is more convenient to compare. The larger this area, the better an algorithm performs. Intuitively speaking, ROCA can be regarded as the expectation of precision w.r.t threshold, assuming the threshold is chosen at random uniformly.

Besides, the previous study finds that the number of leaf nodes is so large that they influence our evaluation. i.e. for a normal user with 400 buddies, there may be 100,000 nodes within two hop. Among the 100,000 nodes, there may be 7,000 nodes belonging to the same community with the observer. Under such situation, a naive algorithm assigning every node to label 0 can reach an accuracy of above 90%. This outcome is of course not interesting at all. To evaluate the current work, we want to limit the scope of training and testing sets(full 2-hop topology is still served to algorithms).

To conclude our evaluation method:

- Use ground truth rather than quality function.
- Each algorithm outputs a "likelihood" value, and we calculate ROCA by varying the threshold of this likelihood value.
- Full 2-hop topology is served to all algorithms, but evaluation is done within a limited scope, like only the level 1 nodes, without degree 1 nodes, etc. Exact choice and justification will be provided in the formal report.

2.2 Spectral Clustering

Although we didn't use the term "spectral", we actually applied one very useful spectral technique, PageRank. To be short, variations of PR are all

the principal eigenvector of a certain matrix. This value is shown to be informative for finding some local sparse cuts[7].

As is known, there are much more eigenvectors that are not used in PR. It's no surprise that those successive eigenvectors can help to distinguish nodes from different clusters. Luxburg's tutorial[6] uses a simple example to illustrate this idea.

Note one thing that although we talk "eigen vectors" in both paragraphs above, they corresponds to different matrix. In calculating PR, the matrix is obtained from (maybe weighted) adjacency matrix. However, in spectral clustering practice, people find the corresponding Laplacian is more useful.

However, taking as many eigenvectors as possible may not be a good idea. For one thing, computational cost is higher when dealing with more eigenvectors. For another thing, not all eigenvectors are informative. Luxburg[6] provides some practical guides on the choice of number of eigenvectors. The eigen gap is taken into consideration as a general rule. However, bearing in mind that the current work is dealing with limited 2-hop topology, there can be proved to be definitely many non-informative eigenvectors. Spielman gives a lemma in his lecture notes[4]: **if two degree one nodes(say, i, j) have a common neighbour, there will be a eigen vector taking the i-th entry as 1 and j-th entry as -1**. In our case, large number of level 2 nodes can form this kind of pair. The computed eigenvectors may be a linear transformation of such standard vectors. The transformed version is also only able to distinguish node by node, carrying no cluster information.

Chanllenges on this line:

- How many eigenvector to use?
- What's the number of clusters k used in the meta clustering?
- How to do postprocessing to give final class labels?
- Which form of Laplacian performs is preferred? (Laplacian, Normalized Laplacian, One-side Normalized Laplacian)
- Any theoretical guarantee for the above issue? (especially in the context of 2-hop)

2.3 Personalized PR

The general form of personalized PR is shown in eqn(1).

$$\vec{v} = \alpha A^{\mathrm{T}} \vec{v} + (1-\alpha)\frac{\vec{b}}{||\vec{b}||_1} \tag{1}$$

If we want to compute the PPR for a set of starting nodes, we only need to set all entries corresponding to those nodes to 1, and others are left zero.

Of course, setting those non-zero entries to values other than 1 is possible but in the current work we only consider 1 for simplicity(meaning no bias on the starting set).

In this work, we'll do a survey on PPR related literature. We seek for potential theoretical characterization of our problem, since in the former work we already find that PPR performs the best alone.

2.4 Budgeted Learning

In practice, resources are often limited, so we want to do budgeted learning. That is, finding the tradeoff between cost and performance.

This research line roots from the observation on PPR:

1. PPR can be linearly combined:

$$\begin{aligned}
\vec{v} &= \alpha A^{\mathrm{T}}\vec{v} + (1-\alpha)\frac{\vec{b}}{||\vec{b}||_1} && (2)\\
&= \alpha A^{\mathrm{T}}\vec{v} + (1-\alpha)\frac{\sum_{i\in\mathrm{support}(\vec{b})} e_i}{||\vec{b}||_1} && (3)\\
&= \frac{1}{||\vec{b}||_1}\sum_{i\in\mathrm{support}(\vec{b})} \alpha A^{\mathrm{T}}\vec{v_i} + \frac{1}{||\vec{b}||_1}\sum_{i\in\mathrm{support}(\vec{b})} (1-\alpha)e_i && (4)
\end{aligned}$$

   where e_i is the unit vector with i-th entry equal to 1, and v_i is the PPR with only i-th vertex in the starting set, namely:

$$\vec{v_i} = \alpha A^{\mathrm{T}}\vec{v_i} + (1-\alpha)\vec{e_i} \quad (5)$$

   and we claim that

$$\vec{v} = \frac{1}{||\vec{b}||_1}\sum_{i\in\mathrm{support}(\vec{b})} \vec{v_i} \quad (6)$$

   This is easy to check by pluging eqn(6) and eqn(5) into eqn(3).

2. Single node PPR can be computed efficiently using Ink Spilling algorithm[7]. The algorithm is theoretically bounded given arbitrary expected precision. Computing all PPR w.r.t single node can be done in the initialization stage.

3. In the former study, the success of PPR tells us that by serving more label 1 nodes, the performance gets better. One extreme case is when all label 1 nodes are served as known nodes to the algorithm. In this case, we expect very high ROCA.

Our application will take the 2-hop topology and some pre-labeled nodes as input. The topology is automatically crawled while pre-labeling induces certain user side cost. Say, the user can label some already known nodes in his community to enhance the algorithm performance. However, manual labeling is time consuming, so we don't want them to label too many nodes.

The study of budgeted learning in our problem aims at finding a small subset of nodes that help enhance performance. The subset can be found in batch mode or in an interactive manner.

2.5 Visualization

Visualization is not the main target in the current work. However, visualization would be a natural by-product when we conducting those experiments mentioned above.

For example, in spectral clustering, we already computed the first several eigenvectors. Spielman has shown that [4] plotting the graph using the 2nd and 3rd eigen vectors of the Laplacian results in very good quality. This 2-D embedding preserves original topology the most and try to stretch the vertices as out as possible.

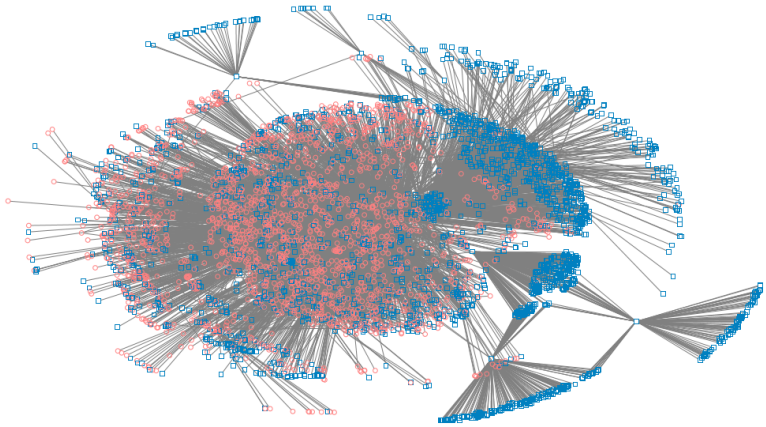


Figure 2: An Old Visualization Using NodeXL

Fig(2) shows one of the visualization result using NodeXL in our former project. It's obvious that the general graph visualization method there does not suit our requirement directly. Besides, NodeXL as a plugin of Excel does not works very efficiently. Our typical graph has more than 100K edges. In order to make NodeXL work, we sample 10K edges uniformly. Taking the physical background into consideration, visualization can be improved in the following way:

- Treat level one nodes and pure level two nodes separately. Plot the latter one insignificantly and focus them around corresponding level

one nodes(note that the NodeXL plotting stretched many such nodes out, which are not interesting).

- Use probably the 2nd and 3rd Laplacian to embed the level one nodes to a 2D plot. Which Laplacian to use is a question. The Laplacian without level two nodes is reasonable.

- Use the spectral clustering result to assign different color to those nodes.

2.6 Cluster Ensemble

This is a practical problem but may be not related with spectral techniques directly. I put here as a memo.

Assume we already have a well performed algorithm that takes the data from one observer as input and output class label. Our intuition is that, by engaging more observers the result has potential to be improved. There are two ways to combine information. The first way is to combine raw data. That is, we first merge the graphs from two observers and try to mine something from the merged graph. This method requires us to tune our already extablished algorithm, since it is originally intended for only one observer. The second way is to do cluster ensemble. By leveraging original algorithm on different observer, we already get some (should be) different partitions. The node sets of different observers may have a certain intersection. [5] first proposed the cluster ensemble problem and use mutual information as the criterion. They formulated an optimization problem to find the ensembled cluster.

3 Challenges

- Preprocessing. Although in our previous project, a lot of preprocessing was done, they're far from enough. With the extension of the this study, other preprocessign may be needed.

- Large computation. Our node number is on the order of 100K. Many direct matrix operations are prohibited. In our former project, we implemented the PR algorithm ourselves in C, which essentially speaking is a practical form of sparse matrix-vector product. This program can give result in several seconds on our data set. However, for spectral clustering, which involves eigen value decomposition, the computational cost is more serious. Current plan comes in two folds:

  1. Deploy other high performance libraries, like CLPACK.
  2. Tailor the raw data. Details need to be designed upon the analysis of the data.

4 Schedule

Table 2: Schedule[2012, April - May]

Time Period	Content	Expected Output
April Week 1-2	PPR	Survey
April Week 2	Ink spilling algo.	Implementation
April Week 3	Budgeted Learning	Sensitivity Simulation
April Week 4	Spectral Clustering	Implementation
May Week 1	Sum up	Report

The current schedule is subject to change according to intermediate outcomes. Initial focus is to do survey, implementation, and simulation. We expect the successful application of those spectral techniques in our problem. Theoretical analysis will be given if there are considerably good results.

References

[1] C.C. Aggarwal. *Social network data analytics*. Springer-Verlag New York Inc, 2011.

[2] Pili Hu. Spectral techniques for community detection on 2-hop topology. GitHub, https://github.com/hupili/Spectral-2Hop, 4 2012. course project of CUHK/CSCI5160.

[3] Pili Hu and Yichao Li. Community detection on 2-hop topology. GitHub, https://github.com/Czlyc/2C-Web-Research, 12 2011. course project of CUHK/CSCI5180.

[4] DA Spielman. Spectral graph theory lecture notes. http://www.cs.yale.edu/homes/spielman/561/, 2011.

[5] A. Strehl and J. Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617, 2003.

[6] U. Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[7] Lap Chi LAU, Spectral Algorithm, `http://www.cse.cuhk.edu.hk/~chi/csc5160/index.html`