Application of Graph Clustering on Scientific Papers Subject Classification

Yutian Liu, Zhefei Yu, Qi Zeng

In this paper, we realize the equivalence between k-means clustering and graph spectrum clustering, and implement a "multi-level algorithm" which combines the advantage of fast computation from k-means and global maximization from spectrum clustering. Then we apply this algorithm on the scientific paper graph which is obtained out of self-defined crawlers over INSPIREHEP databse and perform the clustering. An evaluation is performed and the reason for the imperfect clustering is discussed.

I. INTRODUCTION

The problem that we are interested in here is the clustering of scientific papers within a database. More specifically, due to the existence of citation and reference, papers are actually closely connected to each other if they belong to a same category (topic, research field, etc.). Furthermore, the concept of "category" really depends on the scale we observe this network. All these network features motivate us to think about papers as a graph object with nodes being the paper and edge being the citation/reference relation. Specifically, we want to use state-of-art graph clustering technique to explore detailed structure of scientific paper graph, aiming at subject classification that they belong to.

II. METHODS USED IN GRAPH CLUSTERING

In a graph $G = (\nu, \epsilon)$ with vertices ν and all edges ϵ, we can define an adjacent matrix to represent it:

$$A_{ij} = \epsilon_{ij} \tag{1}$$

where ϵ_{ij} is the weight of edge connecting vertex i and j. Notice that in general A_{ij} is not a symmetric matrix because we don't expect the link to be directed in general. Based on this, we can then further define a link function between two clusters:

$$links(\mathcal{A}, \mathcal{B}) = \sum_{i\in\mathcal{A}, j\in\mathcal{B}} A_{ij} \tag{2}$$

which represent the connection between two subset within a graph. And thus we can define the degree of a subset $\mathcal{A}$ to be $degree(\mathcal{A}) = links(\mathcal{A}, \nu)$ where ν represents all vertices.

In analog to minimizing objective function $\sum_c \sum_{i\in c} ||P_{i_c} - \mu_{i_c}||^2$ as we did in k-means clustering algorithm, one also needs to define a proper objective function in graph clustering. There are many different but quite similar definition of objective function used in graph clustering and we can write them in a general form called weighted association as following:

$$WAssoc(G) = max_{\nu_l \ldots \nu_k} \sum_{c=1}^{k} \frac{links(\nu_c, \nu_c)}{\omega(\nu_c)} \tag{3}$$

where ν_l is each cluster and $\omega(\nu_c) = \sum_{i\in\nu_c} \omega_i$ (ω_i is the weight of each vertex).

A. Spectral Method

To do the maximization/minimization of the objective function defined above, we still needs to do a lot of derivation to transform it into a problem of maximizing the trace of a quadratic form. And then, from linear algebra, we only need to perform a diagonalization of the matrix in the middle and pick those k eigenvectors with descending eigenvalue order. Due to the limitation of pages, we have to omit detailed mathematical derivation but one could refer to reference (will be appended in the final project summary). Specifically, we have eventually

$$WAssoc(G) = max_Y trace(Y^T W^{-1/2} A W^{-1/2} Y) \tag{4}$$

where A is adjacent matrix, W is a matrix formed by our weight function and Y is related to indicator vector ($y_c(i) = 1$ if i is in cluster c).

It turns out spectral method is very powerful in graph clustering in that it is guaranteed to reach global extreme point by principle. However, since the eigenvectors are not indicator vectors in general, we have to do some rounding here. One big problem for the spectral method is that it will be very expensive on a large graph simply because of the complexity of solving eigen-problems. As we will show, though spectral method is the optimal one when node size not too size, we have to resort to other techniques when our graph is too large.

B. Equivalent Weighted Kernel k-means Method

One way to tackle this problem is utilizing the equivalence between k-means method and spectral clustering which is delivered and proved by Inderjit S. Dhillon etc. in our reference. The k-means method first needs to be generalized to "weighted kernel k-means" as changing objective function to be

$$\mathcal{D} = \sum_{c=1}^{k} \sum_{a_i\in\pi_c} \omega_i ||\phi(a_i) - m_c||^2 \tag{5}$$

where $m_c = \dfrac{\sum_{a_i\in\pi_c} \omega_i \phi(a_i)}{\sum_{a_i\in\pi_c} \omega_i}$. Here we map our data from original feature space to a hyper-space through function $\phi(x)$. If we plug m_c in and expand all quadratic form, we will find that only inner product between $\phi(a_i)$ remains and thus we can simply replace it with kernel.

The proof of equivalence between weighted kernel k-means objective function and graph clustering objective function basically only involves several algebra tricks, though it is not intuitive to build such connection. First we define a $n \times k$ matrix Z

$$Z_{ic} = \frac{I\{a_i \in \pi_c\}}{s_c^{1/2}} \tag{6}$$

where $s_c = \sum_{a_i \in \pi_c} \omega_i$. Realizing that columns of Z is mutually orthogonal, we can then get that

$$\mathcal{D} = \sum_{i=1}^{n} \omega_i ||\Phi_i - (\Phi W Z Z_i^T)||^2 \tag{7}$$

where $\Phi_i = \phi(a_i)$ and W is diagonal matrix with weights on it. Then we can get

$$\mathcal{D} = ||\Phi W^{1/2} - \Phi W^{1/2} \widetilde{Y} \widetilde{Y}^T||_F^2 \tag{8}$$

where $\widetilde{Y} = W^{1/2} Z$ and $||A||_F^2$ is the Frobenius norm of a matrix. With the feature that $||A||_F^2 = trac(A^T A)$ and some algebra, one will eventually get

$$\mathcal{D} = trace(W^{1/2} \Phi^T \Phi W^{1/2}) - trace(\widetilde{Y}^T W^{1/2} \Phi^T \Phi W^{1/2} \widetilde{Y}) \tag{9}$$

and thus

$$min_Y \mathcal{D} = max_Y trace(\widetilde{Y}^T W^{1/2} K W^{1/2} \widetilde{Y}) \tag{10}$$

By comparing this to the objective function defined in graph clustering, the spectral method and k-means turns out to be equivalent by doing some proper transformation between adjacent matrix and kernels. One subtlety here is that a kernel transformed from adjacent matrix is not necessarily a valid kernel. To do so, it is proposed in the paper that we can “boost” our obtained kernel to be

$$K' = \sigma W^{-1} + W^{-1} A W^{-1} \tag{11}$$

where σ is a parameter that should be large enough to make K' positive-definite. Further, in case A is not symmetric, one can replace it with $\dfrac{A^T + A}{2}$ and everything is just the same under the context of taking trace. Actually, as we try in our experiment, the choice of parameter σ is very subtle in that it cannot be too large in order to avoid sensitivity to the local minimum while it should make kernel valid at the same time.

By such equivalence, in case the graph is super large and running it with spectral method is expensive, we can simply transform it to a k-means problem which is much faster to compute. Conversely, since k-means algorithm is sensitive to the local minimum (and initial clusters), we can also transform a k-means problem into a spectral clustering problem to achieve optimization when the computation is affordable.

C. Multi-level Strategy

Multi-level strategy is an approach proposed in the reference to combine advantages of speed in k-means algorithm and global optimization in spectral method. This strategy can be divided into three steps: coarsening, base clustering and refinement:

1. Coarsening. First, we need to coarsen our graph into a much smaller graph with nodes size only about one order larger than number of clusterings. This could be done level-by-level. In each level, for some vertex x, we look for the unmarked vertex y that maximize $\dfrac{edge(x,y)}{\omega(x)} + \dfrac{edge(x,y)}{\omega(y)}$ and merge them (and mark them afterward). When all points are marked, we move to next level with merged graph as input and them do all the iterations.

2. Base clustering. When the graph finish coarsening, we run standard spectral clustering on it. Since graph size is small now, the spectral clustering is much efficient.

3. Refinement. Then we just follow the same path down to original graph level-by-level. In each level, we release points merged in this level and assign them to the same cluster as merged points initially. Then we run the equivalent k-means clustering on this refined graph. Since we have a very optimized initial clustering here, the result will be much less sensitive to local minimum.

III. DATA ACQUISITION

Since we cannot find a prepared good enough data (especially with tags), we decide to obtian the data by ourselves through webpage crawler. The database we use is “inspirehep.net” which is a compiled database primarily for papers in high-energy physics. One great advantage of using this database is its closure – Most of the citation/reference of each paper is also in this database.

It turns out the web page crawling is much more challenging than what we expected originally. So far we have only been able to implement single crawler that starts from an initial paper and crawls down the citation using breadth-first search. The maximal depth is severely restricted due to the bandwidth and time (if we sends out too many concurrent http request, the connection lost rate will increase a lot, or even worse, rejected by the server). What is worse, data crawled by single crawler will be just a tree structure rather than graph structure.

First, we start from some paper with a reasonable citations. Then we look for all citation links and do a breadth-first search. During the search, paper with too low citation (< 10). Also, during the search, crawler will collect seed-papers for next round of crawling. The seed-papers should have at least 100 citations and a different tags (a standard subject classification given by database) from initial paper of current crawling. When crawler reaches maximal depth set by user, it will jump to a random paper within seeds collected in this round and then do the whole crawling again starting there. Such whole procedure is repeated until number of iteration reaches maximal value set by user or there is no seed left.

It turns out this routine works to some extend, as we can see in the figures below. However, we can still infer from the graphical structure that this is a very typical tree structure in that we can see clear boundaries between levels. Furthermore, this method costs too much time in crawling, which makes the data acquisition unrealistic when one wants to get a larger sample of the network or when some "star paper" has thousands of citations. One way to treat this is doing a random sampling throughout the crawling. Explicitly speaking, when one is doing the width-first searching of citations, once the collection of next level is established, rather than crawling them over, one pick a random sub-sample and only crawls that part. This simple trick turns to be very powerful in getting a reasonable citation graph.

Figure 1 is a comparison between two graphics with and without this random sampling procedure. The difference is very obvious. Without random sampling, links between articles are actually pretty low. Articles within one level basically are independent with each other. The ratio between links and nodes is about 2 : 1, meaning this is basically just a binary tree structure. On the other hand, when random sampling is adopted, the link-to-node ratio becomes about 7.2 : 1 which is pretty reasonable. From the graph we can also see that the connections increase significantly. Therefore, this one is fixed as the data we use in the following analysis. The total number of nodes is 1171 and total number of edges is 8473. For test purpose, we also include the tag given by the website and there are in total four labels for our sample. More details will be discussed immediately below.

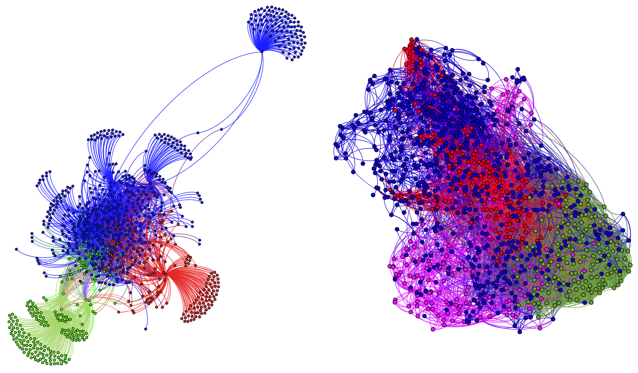
FIG. 1: Left: Sample graph without random sampling; Right: Sample graph after random sampling

IV. EXPERIMENTS ON CRAWLED CITATION GRAPH

In the INSPIREHEP database, there is already a labeling system which assigns a certain category to each paper. Such classification is based on the whole understanding in physics and thus we believe it is a "truth reference" in our test. In the data sample we crawled, there are in total four labels: "hep-ph" (high energy physics phenomenology), "hep-exp" (high energy physics experiment), "hep-th" (high energy physics theory) , "astro-ph" (astrophysics, including cosmology) and "X" category standing for unknown category which represents error in the website scraping. After the data is set up, we are able to apply our algorithms on it and see the effect of clustering.

A. Initialization

As we discussed earlier, the multi-level algorithm design is to avoid the great uncertainty brought by k-means method itself. After a multi-level coarsening process and a spectrum clustering at the most simply graph, we have actually already obtained a roughly correct guess of the final clustering, which make the further equivalent k-mean clustering in refinement steps much more easier and reliable. To see this effect of initialization directly, we do such a comparison between multi-level and k-mean algorithm initialized state: For k-means, it is just random assignment; For multi-level algorithm, it is the state after coarsening and base-clustering. The comparison plot is shown in Figure 2. The graph here is the data from Facebook (downloaded from "snap.stanford.edu") because we want to test the initialization effect independently of sample used.

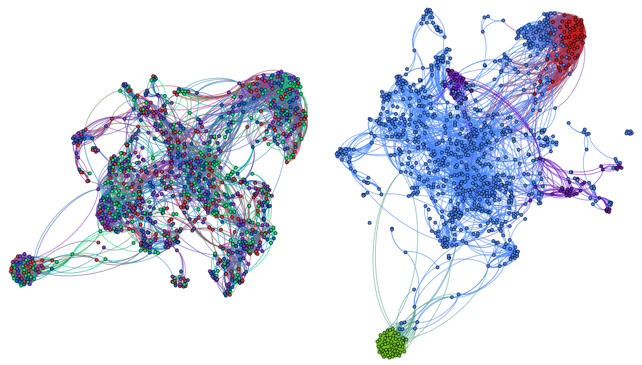
FIG. 2: Left: initialized state of k-mean; Right: initialized state after base-clustering

As we can see, in the normal k-means clustering, random category assignment is pretty bad. As we will show later, such huge uncertainty results in significant unreliability of k-means method. For the multi-level design, it is much more robust in that the spectrum clustering actually have done some clustering here.

B. Level Decision

Another issue in multi-level algorithm is the determination of levels in coarsening (and thus refinement). In our reference paper, a criterion of stopping coarsening is delivered. However, when we test this on our sample, we realize that this method might not in general apply to any problem. Initially we get 10 levels and this gives a very low objective function value (29.94). Therefore, we manually change the coarsening level to 5 (55.27).

C. Comparison between Multi-level and Weighted K-means Clustering

Now, let us take a look at the final result of clustering. First we could make a comparison between multi-level and k-means algorithm. We set the number of clustering to be 4 and below Table I is a chart showing the fraction of articles subscribed to each category (sorted from highest percentage to lowest one):

Category	Multi-level	k-means	random	truth
1	62.77%	29.63%	26.56%	74.81% (hep-ph)
2	13.41%	29.21%	25.28%	10.33% (hep-ex)
3	12.04%	27.92%	24.42%	9.56% (hep-th)
4	11.78%	13.24%	23.74%	1.37% (astro-ph)

TABLE I: Clustering and Truth Tag Percentage Distribution

Note, there is an additional component in truth tag which is around 1.37%. This part represents unknown category out of the flaws in crawlers. Since they are very small we will simply neglect them.

From this table, we can see that multi-level algorithm is way much better than k-means in that it can better reproduce the percentage distribution in truth tags. Furthermore, we put a column called "random" which is generated by random clustering for further comparison. Similarly, we can also make a comparison table of objective function as shown in Table II.

	multi-level	k-means	random
Objective Function	55.27	46.04	14.31

TABLE II: Objective Function

From this table, in terms of objective function maximization, multi-level algorithm does the best while k-means method is actually not that bad – at least it is still far from random clustering. Therefore, it is unfair to assert that k-means is effectively random clustering. But it is true that, as mentioned in the initialization part, a random initialization in k-means severely degrades its performance, especially in such a complicated graph clustering application.

The visualization of two clustering results can be found in Figure 3 which is consistent with statement above.

D. Comparison between Multi-level and Spectrum Clustering

Now, we turns to comparison between multi-level and spectrum clustering. The percentage distribution is shown in Table III.

Category	Multi-level	Spectrum	truth
1	62.77%	63.88%	74.81% (hep-ph)
2	13.41%	21.69%	10.33% (hep-ex)
3	12.04%	7.86%	9.56% (hep-th)
4	11.78%	6.58%	1.37% (astro-ph)

TABLE III: Percentage Distribution

Both multi-level and spectrum clustering gives pretty similar result to either each other or the truth tags, which is good because spectrum clustering is guaranteed to reache global maximization in principle. The table on objective function is Table IV.

	multi-level	spectrum
Objective Function	55.27	55.65

TABLE IV: Objective Function

The objective function table further confirms that multi-level and spectrum basically gives the same result.

E. Multi-level Evaluation

The canonical way of doing evaluation is the estimation of efficiency and fake-rate. These could be inferred by looking at the clustered category distribution within one truth tags. For example, we can see the truth tag "hep-exp" and see how many of them is really assigned to the correct category in clustering. Similarly, we can also do this in the inverse way – one looks at the clustering and estimate the distribution of truth tags within each clustering. Both way can give us a sense how the clustering is doing in an equivalent way.

A summary of the distribution of truth tags within each multi-level clustering is shown in the Table V.

From this table, one can see that each multi-level clustering is dominant by the "hep-ph" articles. Considering that "hep-ph" papers are dominant in our sample, this is not so surprising though it is definitely not good. What

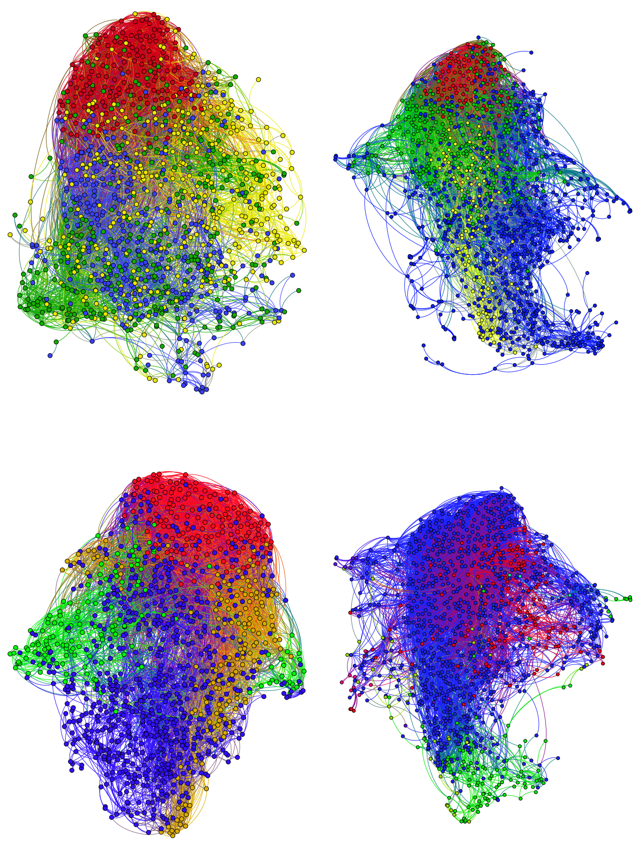

FIG. 3: LeftTop: K-means; RightTop: Spectrum; LeftBottom: Multi-level; RightBorrom: TruthTag

Multi-level Clustering	hep-ph	hep-exp	hep-th	astro-ph
1	524	56	105	43
2	132	11	6	3
3	130	8	0	0
4	90	46	1	0

TABLE V: Summary Table of Clustering

is worse, for each truth tag, the dominant population also is concentrated in multi-level tag "1". All these means that our clustering is not very ideal because it cannot really discriminate between these four categories.

One of the most important reason responsible for this is the data sample we are using. Frankly speaking, the graph that we are using is still not complete, in the sense that it is not a good representation of the graph structure. For example, according our crawler rules, the truth tag between different iteration is not the same. We use this rule because we want to get a sample with diverse possibilities. However, this apparently bias the whole structure because we know the original structure in this local region should be dominant by the "hep-ph" paper while we pick several connected local regions with different truth tag. Statistically speaking, we are picking statistical fluctuation rather than the dominant structure itself. But due to the limit on our resources, it is impractical for us to really get a complete enough graph sample this moment.

If we go one step further, we could infer that the clustering algorithm that we use might not be suitable for a sample with strong statistical fluctuation. To show this, we can take a look at the objective function out of each clustering algorithm and the truth tag itself in the Table VI.

	K-means	spectrum	multi-level	truth tag
Objective Function	46.04	55.65	55.27	24.37

TABLE VI: Summary Table of Objective Function

It's surprising to see that all clustering method actually reaches a objective function value much larger than the one obtained from truth tag itself. This implies that we might be choosing an inappropriate objective function whose maximum does not correspond to the actual solution to our problem.

V. CONCLUSION

Eventually we conclude that, though the graph clustering algorithems perform well on maximizing the objective function, they do not work very well on categorizing the sample scientific papers. It is because the basic Ratio association/cut may not be the best objective function for this problem. Also, the sample we collected is biased by our crawling routine.

[1] N. Cristianini, J. Shawe-Taylor, and J. Kandola, *"Spectral Kernel Methods for Clustering"*, Proc. 14th Advances in Neural Information Processing Systems, 2001.

[2] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis, *"Weighted Graph Cuts without Eigenvectors: A Multilevel Approach"*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 29, No. 11, 2007

[3] H. Zha, C. Ding, M. Gu, X. He, and H. Simon, *"Spectral Relaxation for k-Means Clustering"*, Proc. Neural Information Processing Systems, 2001.

[4] I. Dhillon, Y. Guan, and B. Kulis, *"Kernel k-Means, Spectral Clustering and Normalized Cuts"*, Proc. 10th ACM Knowledge Discovery and Data Mining Conf., pp. 551-556, 2004.