A LETTERS JOURNAL EXPLORING THE FRONTIERS OF PHYSICS

EPS · Società Italiana di Fisica · edp sciences · IOP Institute of Physics

LETTER

Community structure detection in complex networks with partial background information

To cite this article: Zhong-Yuan Zhang 2013 *EPL* **101** 48005

H_1: How to improve CD when community structure is fuzzy.

H_2: Not really explored.

H_3:– Add pairwise constraints;
– Create new adj matrix B where (i,j) = α if must link (i.e. known same community); (i,j) = β if must not link (i.e. known separate) and all others retaining their old value.
– Run a CD algo on new matrix (they used SC, NMF, INFOMAP).

Complexity:- N/A, no CD algo.
– The matrix recalculation will be at least O(N) if just a label lookup, but could be much higher if calc other statistical or parametric similarity measures.

Datasets:
GN, LFR, KARATE, FOOTBALL

Thoughts
(1) Assumes K is known a-priori; Brute force K to maximize mod of nets.
(2) This approach (like many others) doesn't handle self-loops. There exist situations self-loops occur (e.g. package repo's) or emails to self.
* Add to "Hard" Data.

They offer an open question "How best to select the nodes to impose constraints on"
↳ Suggest that they use some stat similarity measure.
↳ or if not imposing on whole matrix, how to pick subset.
↳ Wouldn't KL's be a good candidate for this?

A Letters Journal Exploring the Frontiers of Physics

February 2013

EPL, **101** (2013) 48005
doi: 10.1209/0295-5075/101/48005

www.epljournal.org

Community structure detection in complex networks with partial background information

Zhong-Yuan Zhang

School of Statistics, Central University of Finance and Economics - Beijing, PRC

received 14 November 2012; accepted in final form 7 February 2013
published online 5 March 2013

PACS 89.75.Hc – Networks and genealogical trees
PACS 89.20.Ff – Computer science and technology

Abstract – Constrained clustering has been well-studied in the unsupervised learning society. However, how to encode constraints into community structure detection, within complex networks, remains a challenging problem. In this paper, we propose a semi-supervised learning framework for community structure detection. This framework implicitly encodes the *must-link* and *cannot-link* constraints by modifying the adjacency matrix of network, which can also be regarded as de-noising the consensus matrix of community structures. Our proposed method gives consideration to both the topology and the functions (background information) of complex network, which enhances the interpretability of the results. The comparisons performed on both the synthetic benchmarks and the real-world networks show that the proposed framework can significantly improve the community detection performance with few constraints, which makes it an attractive methodology in the analysis of complex networks.

Introduction. – Evidences have shown that there are often modules or community structures in complex networks [1]. For example, a community could be a set of proteins that have similar functions in a protein-protein interaction (PPI) network, or it could be a group of fans that like visiting similar kind of music web pages, or a university club, etc. Though there is still no standard and clear definition of community structure, we may regard a community in complex networks as a set of nodes that have similar link-pattern, or in other words, these nodes have similar preference and connect to the other nodes in a similar way. The most common and widely studied community is a subgraph that is densely interconnected but loosely connected with the rest of the graph. Meanwhile, there are also other types of communities. Discovering communities is very important for revealing the organization and the functions of the network, such as understanding how the units in some systems communicate with each other and work together, or learning how the new ideas or diseases spread in a group of persons [2], etc.

How to detect community structures has thus become a hot topic, and many interesting models and algorithms have been developed and have achieved good results. But all of these methods are in essence a kind of unsupervised learning, meaning that they only make use of the network topology information. However, in many real scenarios, there is usually some background information that could also be used in detecting the communities. This information can be treated as additional constraints, and how to combine the information with the network topology to guide the detecting process is an interesting problem that is worthy of working on.

In this paper, we propose a semi-supervised framework to incorporate prior information into community structure detection. Our framework is flexible to integrate various known information. One can easily provide pairwise constraints on a few nodes in the network, specifying whether they must or cannot be in the same community structure, based on the background information and domain knowledge. For example, the nodes that have similar functions should be must-link, or the nodes that have different opinions should be cannot-link. The framework implicitly encodes the must-link and cannot-link constraints by modifying the adjacency matrix of the network, which can also be regarded as the de-noising process of the consensus matrix of the community structures, *i.e.*, creating connections within communities and removing connections across communities.

Semi-supervised learning for community structure detection. – In this section, we formulate our semi-supervised framework for community structure detection. Firstly, we introduce the definition of adjacency matrix

$A^{[0]}$ of an undirected and unweighted simple graph G with n nodes:

$$A^{[0]}_{ij} = \begin{cases} 1, & \text{if } i \sim j, \\ 0, & \text{if } i = j \text{ or } i \nsim j, \end{cases}$$

where $i \sim j$ means there is an edge between node i and j, and $i \nsim j$ means there is no edge between them. Here $A^{[0]}$ is $n \times n$ and symmetric.

Note that the diagonal elements of $A^{[0]}$ are all zeros, but these zeros are obviously different from the ones at the off-diagonal positions which mean there are no connections between the nodes. Hence we here set the diagonal elements of $A^{[0]}$ to 1. The revised adjacency matrix is denoted by $A^{[1]}$. Another variation of $A^{[0]}$ is its complementary matrix $C^{[A]} = 1 - A^{[0]}$.

Incorporating prior knowledge into adjacency matrix. In many real applications, we often have some background information that can be used for community structure detection. Specifically, we consider the following two types of pairwise constraints: 1) `Must-Link constraints` C_{ML}: $(i,j) \in C_{ML}$ means that the two nodes i and j must belong to the same community; 2) `Cannot-Link constraints` C_{CL}: $(i,j) \in C_{CL}$ means that the two nodes i and j cannot belong to the same community.

We incorporate the constraints C_{ML} and C_{CL} into the adjacency matrix $A^{[1]}$ to get a new matrix B as follows:

$$B_{ij} = \begin{cases} \alpha, & \text{if } (i,j) \in C_{ML}, \\ 0, & \text{if } (i,j) \in C_{CL}, \\ A^{[1]}_{ij}, & \text{otherwise}, \end{cases} \tag{1}$$

where α is a positive constant.

As one can see, if we set α to 1, and for all the pairs of nodes, we know whether they should belong to C_{ML} or C_{CL}, or in other words, we know very well the community structures in the graph, the adjacency matrix will reduce to the standard consensus matrix, whose (i,j)-th element means whether node i and node j are in the same community, 1 means yes and 0 means no. Hence from the viewpoint of consensus matrix, incorporating prior knowledge can be regarded as the de-noising process.

We have tried different α, *i.e.*, $\alpha = 1$ and $\alpha = 2$, and the results of $\alpha = 2$ always get better. We omit the comparisons here due to space limit.

After incorporating background information into the adjacency matrix, we then apply non-negative matrix factorization (NMF), spectral clustering and InfoMap, which are of the most common and widely used models in unsupervised learning, for community structure detection.

Non-negative matrix factorization (NMF, [3–6]). NMF can be expressed as follows: given a non-negative objective matrix X of size $n \times m$, columns of which are samples and rows are features, we try to find two non-negative matrices F of size $n \times k$ and G of size $m \times k$ such that $X \approx FG^T$. This problem is often formulated as the following nonlinear programming:

$$\begin{aligned} \min_{F,G} \quad & J(X \| FG^T), \\ s.t. \quad & F \geqslant 0, \quad G \geqslant 0, \end{aligned} \tag{2}$$

where $J(X \,\|\, FG^T)$ is the cost function that measures the dissimilarity between X and FG^T, and $\geqslant 0$ means that F and G should not have negative entries. The most popular algorithm designed for NMF is multiplicative update rules. The objective matrix X for NMF can be selected as B.

In [7], it showed that the diffusion-kernel–based similarity matrix SK[1] was the best choice for the objective matrix X among all the candidates, hence we also tested the performance of SK in this paper.

The community structures of the network can be obtained from G: node i is of community k if G_{ik} is the largest element in the i-th row of G.

1) Standard NMF with least-squares error: If $J(X \,\| FG^T)$ is selected as the least-squares error: $J(X \,\| FG^T) = \| X - FG^T \|^2_F$, the algorithm of multiplicative update rules can be summarized in Algorithm 1. In this paper, the iteration number `iter` is set to 100.

Algorithm 1 Non-negative Matrix Factorization (Least-Squares Error)

Input: X, iter.
Output: F, G.

1: **for** $t = 1$: iter **do**.
2: $F_{ik} := F_{ik} \dfrac{(XG)_{ik}}{(FG^TG)_{ik}}$.
3. $G_{ik} := G_{ik} \dfrac{(X^TF)_{ik}}{(GF^TF)_{ik}}$.
4. **end for**.

2) Standard NMF with K-L divergence: If $J(X \,\| FG^T)$ is selected as the K-L divergence: $J(X \| FG^T) = \sum_{i,j}[X_{ij} \log \frac{X_{ij}}{(FG^T)_{ij}} - X_{ij} + (FG^T)_{ij}]$, the corresponding update rules of F and G are

$$F_{ik} := \frac{F_{ik}}{\sum_j G_{jk}} \sum_j \frac{X_{ij}}{(FG^T)_{ij}} G_{jk};$$

$$G_{jk} := \frac{G_{jk}}{\sum_i F_{ik}} \sum_i \frac{X_{ij}}{(FG^T)_{ij}} F_{ik}.$$

[1]Definition of diffusion kernel K and the similarity matrix SK [7,8]: $K = \lim_{n\to\infty}(1 + \frac{\beta L}{n})^n = \exp(\beta L)$, where L is the opposite Laplacian of $A^{[0]}$:

$$L_{ij} = \begin{cases} 1, & \text{if } i \sim j, \\ -d_i, & \text{if } i = j, \\ 0, & \text{otherwise}, \end{cases}$$

and d_i is the degree of node i; $SK_{ij} = \frac{K_{ij}}{\sqrt{K_{ii}K_{jj}}}$. We set $\beta = 0.2$ in this paper. Note that there is a MATLAB command "expm" for the exponential of a matrix.

3) Symmetric NMF (SNMF): There is a variant of NMF for semi-supervised clustering, whose objective function can be formulated as: $\| X - GSG^T \|_F^2$. The update rules of G and S are [5]

$$G_{ik} := G_{ik} \frac{(XGS)_{ik}}{(GSG^TGS)_{ik}};$$
$$S_{ik} := S_{ik} \frac{(G^TXG)_{ik}}{(G^TGSG^TG)_{ik}}.$$

4) Bayesian NMF [6]: It optimizes the NMF model under the Bayesian framework, and can get better results under some circumstances.

Spectral clustering [9]. Spectral clustering is very powerful in its simplicity and effectiveness, which can be summarized in Algorithm 2. Note that there are many variations of the standard one, and the detailed analysis can be found in [10,11].

InfoMap [12]. This model grew out of information theory, and tries to reveal the communities by optimizing a quality function about the minimum description length of random walks on the network. The model is among the best for community detection [13].

Algorithm 2 Spectral Clustering

Input: $B \in \mathbb{R}^{n \times n}$.

Output: Community Label $Y \in \mathbb{R}^{n \times 1}$ of the n nodes.

1: $L = D^{1/2}BD^{1/2}$, where D is the diagonal matrix with the element $D_{ii} = \sum_j B_{ij}$.
2: Formimg the matrix $X = [x_1, x_2, \cdots, x_k] \in \mathbb{R}^{n \times k}$, where $x_i, i = 1, 2, \cdots, k$ are the top k eigenvectors of L.
3: Normalizing X so that rows of X have the same L_2 norm: $X_{ij} = X_{ij}/(\sum_j X_{ij}^2)^{1/2}$.
4: Clustering rows of X into k clusters by K-means.
5: $Y_i = j$ if the i-th row is assigned to cluster j.

An illustrative example. We close this section by an illustrative example as follows: we try to detect the community structures in a GN network with 128 nodes. (For details, see the "Data description" subsection, $Z_{out} = 10$.) The network has 4 communities with 32 nodes each. The heatmap of the corresponding adjacency matrix $A^{[1]}$ is shown as the leftmost panel in fig. 1. If we have prior knowledge about the network structure so that we can determine a percentage of pairs of nodes as must-link or cannot-link, we can incorporate them into $A^{[1]}$. As one can see in fig. 1, the adjacency matrix becomes more and more clear as the percentage of pairs constrained increases, and finally reduces to the standard consensus matrix of the community structures. This example demonstrates that background information is valuable to improve the accuracy of community structure detection.

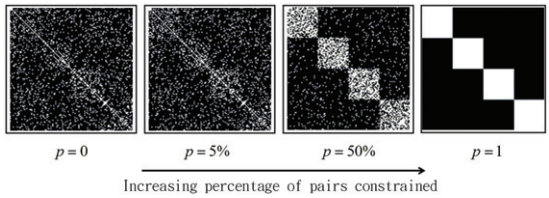


Fig. 1: An illustrative example to show the process of incorporating prior information into the adjacency matrix as denoising the consensus matrix.

Experimental results. – In this section, we empirically demonstrated the effectiveness of our proposed semi-supervised framework for community structure detection by applying NMF, spectral clustering and InfoMap with the de-noised consensus matrices to several well-studied networks.

Data description. We used both synthetic and real-world networks to test the effectiveness of our methods. The details of these datasets are as follows:

1) GN [1]: Maybe the most widely used benchmarks are GN networks. The network has 128 nodes which are divided into four non-overlapping communities with 32 nodes each. The degree of each node is $Z_{in} + Z_{out} = 16$, in other words, each node averagely has exactly 16 edges which randomly connect Z_{in} nodes in its own community and Z_{out} nodes in other communities. As one can see, with the increasing Z_{out}, the community structures will become less clear and the problem more challenging. In this paper, we set Z_{out} to 8.

2) LFR [14]: Indeed, in most of the real applications, the community structures are more complicated than GN networks. The size of the network might be larger, or the numbers of the nodes in different communities might not be identical, or different nodes might have different positions, *i.e.*, some are superstars or hubs and should have higher degrees while the others are leaves. The LFR benchmark networks are thus proposed to address these problems. In LFR networks, both the degree and the community size distributions are power laws, with exponents γ and β, which is more practical. Each node has a fraction $1 - \mu$ of its links with the nodes in its own community and a fraction μ with the other ones. Here μ is called the mixing parameter. We set the parameters of the LFR network as follows: the number of nodes was 1000, the average degree of the nodes was 20, the maximum degree was 50, the exponent of the degree distribution γ was 2 and that of the community size distribution β was 1, and the mixing parameter μ was 0.8. The communities were non-overlapping.

3) Karate [15]: this dataset contains the network of friendships between 34 members of a karate club at an American university. This club was by chance split

into two smaller ones due to the divergence of opinions about the club fees.

4) Football [1]: this dataset contains the network of American football games (not soccer) between Division IA colleges during regular season Fall 2000. There are 115 nodes representing the football teams while an edge means there was a game between the teams connected by the edge. The teams were divided into 12 conferences, and all teams except few (mainly in two conferences) played against the ones in the same conference more frequently than those in other conferences.

Assess standards. In our experiments, the normalized mutual information (NMI, [16]) was used as the standard to evaluate the community structure detection performance. The value can be formulated as follows:

$$\mathrm{NMI}(M_1, M_2) = \frac{\sum_{i=1}^{k}\sum_{j=1}^{k} n_{ij} \log \frac{n_{ij} n}{n_i^{(1)} n_j^{(2)}}}{\sqrt{\left(\sum_{i=1}^{k} n_i^{(1)} \log \frac{n_i^{(1)}}{n}\right)\left(\sum_{j=1}^{k} n_j^{(2)} \log \frac{n_j^{(2)}}{n}\right)}},$$

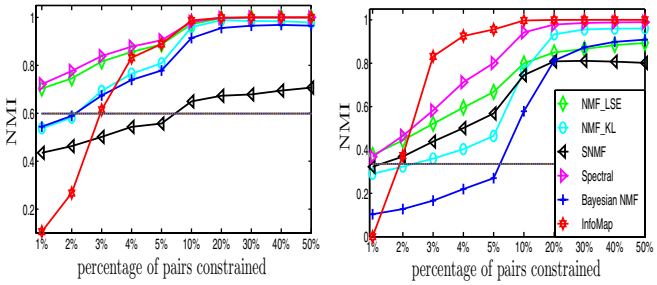
where M_1 is the ground-truth cluster label and M_2 is the computed cluster label, k is the community number, n is the number of nodes, n_{ij} is the number of nodes in the ground-truth cluster i that are assigned to the computed cluster j, $n_i^{(1)}$ is the number of nodes in the ground-truth cluster i and $n_j^{(2)}$ is the number of nodes in the computed cluster j, log is the natural logarithm.

Compared with simply counting the number of misclassified nodes, NMI is more informative, especially suitable for imbalanced datasets (*i.e.*, the numbers of the nodes in different communities are not identical). For example, in a four-sample toy data, the ground-truth cluster label could be 1,1,1,2. The computed cluster labels of two different models were 1, 1, 1, 1 and 1, 1, 2, 2, respectively. In other words, the smaller cluster was masked and not detected by the first model, hence the second model should be better though it also had one sample mis-clustered. But the accuracy (number of misclassified nodes divided by the number of nodes in the graph) results of these two models were all 75%, which was misleading. On the other hand, the NMI under this case was 0 (the numerator of NMI was: $3\log\frac{3\cdot4}{3\cdot4} + 1\cdot\log\frac{1\cdot4}{1\cdot4} = 0$) and 34.56%, respectively, which was relatively more reasonable and informative.

In the case study, we also used the modularity function Q [17,18] as the standard to determine the best community number k. The function can be defined as follows:

$$Q = \sum_{C_k}\left[\frac{L(V_{C_k}, V_{C_k})}{L(V,V)} - \left(\frac{L(V_{C_k}, V)}{L(V,V)}\right)^2\right],$$

where C_k is the k-th community in the graph, $L(V_1, V_2) = \sum_{i\in V_1, j\in V_2, i\neq j} a_{ij}$, and a_{ij} is the element of $A^{[0]}$.



(a) GN, $Z_{out} = 8$, $\alpha = 2$. (b) LFR, $\mu = 0.8$, $\alpha = 2$.

Fig. 2: (Color on-line) Averaged NMI of different models for different percentages of node pairs constrained on GN and LFR datasets. The black horizontal line is the best NMI result that had ever got by NMF_LSE, NMF_KL, SNMF, Bayesian NMF, spectral clustering and InfoMap with no prior knowledge available. "LSE" means least-squares error, "KL" means K-L divergence, "SNMF" means symmetric NMF.

The larger the values of NMI and Q, the better the graph partitioning results.

Firstly, we compared the clustering performance of NMF-based models with different similarity measures including $A^{[0]}$, $A^{[1]}$, $C^{[A]}$ and SK. The results show that $A^{[1]}$ is a competitive one, though there is no single winner. Note that calculating the diffusion kernel is time consuming for large scale networks, hence we used $A^{[1]}$ for the NMF-based models in the following experiments. The details are omitted here due to space limit.

Results analysis. In this subsection, we systematically compared the results of NMI obtained by the models on the artificial datasets and the karate network with prior knowledge available. For an undirected network with n nodes, there are totally $n(n-1)/2$ node pairs available. We randomly picked out some pairs of nodes, and determined whether they belonged to C_{ML} or C_{CL}: if the two nodes had the same community label, they were must-link, otherwise, they were cannot-link. The results were averages of ten trails and given in fig. 2 and table 1. From these figure and table, one can observe that: i) the trends of all the models are generally identical and the values of the averaged NMI increase with the increasing percentage of pairs constrained; ii) for synthetic datasets: GN and LFR, the model of InfoMap and the spectral clustering are better than the NMF-based models, especially for the LFR datasets; iii) for the karate network, NMF with least-squares error performs better; iv) our proposed framework is flexible and model independent, or in other words, it can be naturally combined with many models, such as NMF, spectral clustering, InfoMap, etc.

In summary, our proposed semi-supervised framework does greatly enhance the results of community structure detection by benefitting from the user provided background information.

A case study: college football network. In this subsection, we used the college football network for a case study,

Table 1: Averaged NMI of different models given different percentages of node pairs constrained on the karate dataset. "P" means percentage of node pairs constrained. The meaning of "LSE", "KL" and "SNMF" is identical with that in fig. 2, and "SP" means spectral clustering.

P \ Models	NMF_LSE	NMF_KL	SNMF	SP
1%	**99.84%**	73.38%	59.53%	90.19%
2%	**98.86%**	73.44%	51.50%	90.19%
3%	**99.67%**	82.86%	54.06%	95.10%
4%	**99.84%**	85.18%	60.96%	96.73%
5%	**99.84%**	89.24%	53.74%	95.10%
10%	**100%**	89.14%	57.91%	**100%**
20%	**100%**	98.37%	56.57%	**100%**

Table 2: Basic information about the abnormal teams that played more frequently against the ones in the other conferences. "T" means the team id, "F" means the times that the team played against the other ones in the same conference or in the other conferences, "S" means the same conference, "O" means the other conferences.

T \ F	S	O	T \ F	S	O
37	0	8	*60*	2	6
43	0	7	*64*	2	7
81	1	10	*70*	3	8
83	1	10	*98*	3	5
91	0	9	*111*	0	11
12	4	6	*29*	0	9
25	3	7	*59*	2	8
51	3	6			

and saw the partitioning results of NMF_LSE given different percentages of pairs constrained. Actually, we also tried spectral clustering and got similar results. Details of spectral clustering are omitted here due to space limit.

The teams were separated into 12 conferences, and most of them played against the ones in the same conference more frequently. However, the teams 37, 43, 81, 83, 91 (in conference *IA Independents*), 12, 25, 51, 60, 64, 70, 98 (in conference *Sunbelt*), 111, 29 and 59 played more frequently against the ones in other conferences. Table 2 lists the basic information about these teams, from which one can observe that three out of five teams in *IA Independents* never played against the ones in the same conference and the other two teams played only once.

Firstly, we tried to determine the community number k. We compared the values of modularity Q at different k, and the function achieved its peak value at $k = 11$. By combining the results of Q values in table 3 with the information in table 2, we set the community number $k = 11$ and the teams in *IA Independents* would be assigned to the other eleven conferences based on the outputs of

Table 3: Values of averaged Q functions of NMF_LSE and spectral clustering. The range of the community number k that we have tried is 8–12. The peak values were achieved at $k = 11$. The meaning of "LSE" is identical with that in fig. 2.

Community number \ Models	NMF_LSE	Spectral Clustering
8	0.5770	0.5932
9	0.5831	0.5927
10	0.5890	0.5942
11	**0.5934**	**0.5978**
12	0.5885	0.5951

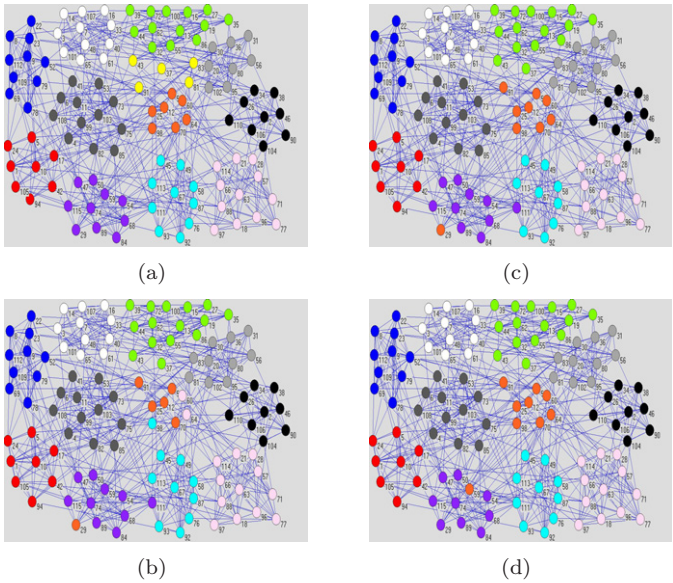


Fig. 3: (Color on-line) Comparison of the results of NMF corresponding to different percentages of pairs constrained. (a) Real grouping in football dataset. There are 12 conferences of 8–12 teams (nodes) each. (b) Outputs of NMF without any prior knowledge. (c) Outputs of NMF corresponding to 5% of pairs constrained. (d) Outputs of NMF corresponding to 20% of pairs constrained.

NMF. Hence there were $115 - 5 = 110$ teams with ground-truth conference labels and totally $110 \times (110 - 1)/2 = 5995$ team pairs available. We randomly selected some pairs as constraints: if the two teams of the pair were in the same conference, they were must-link (ML), otherwise, they were cannot-link (CL).

Figure 3 gives the resulting partitions of NMF corresponding to different percentages of pairs constrained. When given no prior knowledge constrained, there were 5 abnormal teams mis-clustered: teams 29, 60, 64, 98, 111; but after randomly given 5% of pairs constrained, the results were significantly improved and only two abnormal teams were mis-clustered: teams 29 and 111. Finally, when given 20%, there was only one team mis-clustered: team 59. From these results, one can see that: 1) NMF

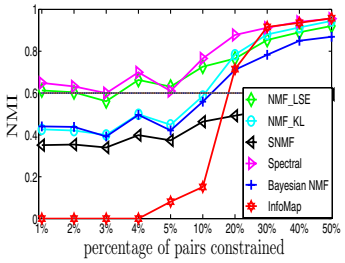


Fig. 4: (Color online) Averaged NMI of different models for different percentages of node pairs constrained on GN datasets ($Z_{out} = 8$). The prior knowledge are given based on rule. The meaning of the black horizontal lines, "LSE", "KL" and "SNMF" is identical with that in fig. 2.

is really good enough in that only some abnormal teams are not correctly clustered; 2) our semi-supervised clustering framework does take the background information and domain knowledge into consideration, which makes the partitioning results more explainable.

How to give the prior knowledge: randomly or based-on-rule. Finally, we ask an interesting question: how to select the prior information and incorporate them into the models? To the best of our knowledge, in practice, the most widely used method is to randomly select some pairs of samples or nodes and manually determine whether they are must-link or cannot-link based on the domain knowledge. But are there any better methods to select the pairs that can either reduce the workload or improve the clustering performance, or both? Indeed, for a large scale network, a very small percentage of pairs may still mean a huge workload. In this subsection, we attempted to introduce a new rule-based method to address this problem. Firstly, we computed the hamming distances between all pairs of nodes (rows of A[1]), and sorted the distances to find the largest and the smallest ones (this step can be finished by programming calculation, not manually). We selected the pairs that have the largest distances and the smallest distances simultaneously. For example, if we wanted to select P pairs of nodes, we selected $P/2$ pairs with the largest distances, and also selected $P/2$ pairs with the smallest distances. Then we manually decided whether the selected pairs were must-link or cannot-link and incorporated them into the clustering process. The results on GN datasets are shown in fig. 4, from which one can observe that our preliminary results are not good enough compared with that of randomly based. Hence we leave the problem open and believe that it deserves further study.

Conclusions and future work. – In this paper, we introduced a semi-supervised community structure detection framework for complex network analysis. The framework adopts a simple strategy to add the supervision of pairwise must-link and cannot-link constraints into the adjacency matrix, which can be regarded as de-noising of the consensus matrix of community structures. The experiments on both the synthetic and real-world networks have demonstrated the effectiveness of the proposed framework. In summary, it can combine the network's functions (background information and domain knowledge) with its topology, making the community structure detection more effective and the results more practical.

We would like to close this paper by raising two interesting problems. Firstly, as we have mentioned above, are there any better methods that can be used for selecting the constraints? A good attempt is the work in [19], which selected the constraints based on various similarity measures, not randomly. Secondly, the proposed framework is very flexible, and can be naturally combined with some other semi-supervised learning models. Researches on this kind of combination are our future working directions.

* * *

This work is supported by the National Natural Science Foundation of China under Grant No. 61203295. The author is very grateful to the reviewers for the valuable comments.

REFERENCES

[1] Girvan M. and Newman M. E. J., *Proc. Natl. Acad. Sci. U.S.A.*, **99** (2002) 7821.
[2] Wu X. and Liu Z., *Physica A*, **387** (2008) 623.
[3] Lee D. D. and Seung H. S., *Nature*, **401** (1999) 788.
[4] Lee D. D. and Seung H. S., *Adv. Neural Inf. Process. Syst.*, **13** (2001) 556.
[5] Chen Y., Rege M., Dong M. and Hua J., *Seventh IEEE International Conference on Data Mining* (IEEE) 2007, pp. 103–112.
[6] Psorakis I., Roberts S., Ebden M. and Sheldon B., *Phys. Rev. E*, **83** (2011) 066114.
[7] Wang R., Zhang S., Wang Y., Zhang X. and Chen L., *Neurocomputing*, **72** (2008) 134.
[8] Kondor R. I. and Lafferty J., in *International Conference on Machine Learning (ICML)* (Morgan Kaufmann Publishers) 2002, pp. 315–322.
[9] Ng A., Jordan M. and Weiss Y., *Adv. Neural Inf. Process. Syst.*, **2** (2002) 849.
[10] Shen H. and Cheng X., *J. Stat. Mech.: Theory Exp.* (2010) P10020.
[11] Ma X. and Gao L., *J. Stat. Mech.: Theory Exp.* (2011) P05012.
[12] Rosvall M. and Bergstrom C., *Proc. Natl. Acad. Sci. U.S.A.*, **105** (2008) 1118.
[13] Lancichinetti A. and Fortunato S., *Phys. Rev. E*, **80** (2009) 056117.
[14] Lancichinetti A., Fortunato S. and Radicchi F., *Phys. Rev. E*, **78** (2008) 046110.
[15] Zachary W. W., *J Anthropol. Res.*, **33** (1977) 452.
[16] Strehl A. and Ghosh J., *J. Mach. Learn. Res.*, **3** (2002) 583.
[17] Newman M. and Girvan M., *Phys. Rev. E*, **69** (2004) 026113.
[18] Newman M., *Proc. Natl. Acad. Sci. U.S.A.*, **103** (2006) 8577.
[19] Ma X., Gao L., Yong X. and Fu L., *Physica A*, **389** (2010) 187.