Community Detection and Link Prediction via Cluster-driven Low-rank Matrix Completion

Junming Shao, Zhong Zhang, Zhongjing Yu, Jun Wang, Yi Zhao and Qinli Yang
Data Mining Lab, University of Electronic Science and Technology of China
{junmshao, qinli.yang}@uestc.edu.cn

Abstract
Community detection and link prediction are highly dependent since knowing cluster structure as a priori will help identify missing links, and in return, clustering on networks with supplemented missing links will improve community detection performance. In this paper, we propose a Cluster-driven Low-rank Matrix Completion (CLMC), for performing community detection and link prediction simultaneously in a unified framework. To this end, CLMC decomposes the adjacent matrix of a target network as three additive matrices: clustering matrix, noise matrix and supplement matrix. The community-structure and low-rank constraints are imposed on the clustering matrix, such that the noisy edges between communities are removed and the resulting matrix is an ideal block-diagonal matrix. Missing edges are further learned via low-rank matrix completion. Extensive experiments show that CLMC achieves state-of-the-art performance.

1 Introduction
Community structure is ubiquitous in real-world networks, and many community detection algorithms have been proposed, e.g., NCut [Shi and Malik, 2000], MCL [Van Dongen, 2000], Modularity [Newman, 2006], Attractor [Shao et al., 2015], FUSE [Ye et al., 2016], ORSC[Shao et al., 2017], MAPPR [Yin et al., 2017], Dcut [Shao et al., 2018], to mention a few. Although community detection has achieved great success during the past decade, finding intrinsic community structure in networks is still challenging. For example, most existing community detection approaches usually work well if (but only if) the communities are well separated. With the increase of inter-cluster edges, the performance of established approaches tends to decrease since they significantly hamper community separation. In addition, many connections in real-world networks are missing. Such problem, further aggravates the difficulty to find high-quality communities. Therefore, identifying inter-cluster edges and supplementing missing edges offer a promising way to enhance the performance of community detection and link prediction simultaneously.

In recent years, matrix completion becomes an important tool in machine learning, and has been extensively studied in recommender systems and computer vision [Liu et al., 2013; Natarajan and Dhillon, 2014; Kang et al., 2016]. Given an observed matrix $A$, matrix completion constructs a new matrix $\hat{A}$ that approximates $A$ at its unobserved entries. Since there are infinite number of matrices that perfectly agree with the observed entries of $A$, some additional assumptions are usually imposed such as low-rank [Cand`es and Recht, 2009; Liu et al., 2013; Hastie et al., 2015; Yang et al., 2018] and sparsity [Lu et al., 2016; Fan and Chow, 2017].

Motivated by aforementioned problems and existing matrix completion techniques, in this paper, we propose a Cluster-driven Low-rank Matrix Completion, called CLMC, which aims to learn an ideal similarity matrix to perform graph clustering and a supplement matrix to conduct link prediction simultaneously in a unified framework. The basic idea of CLMC is to decompose the original data matrix into three additive matrices: the first corresponds to an ideal clustering matrix (Figure 1(b)), which is expected to be a block-diagonal matrix where the inter-cluster edges are removed and meanwhile, the missing or trend connections are supplemented in each community. The second corresponds to the inter-cluster edges, which is viewed as a noisy matrix (Figure 1(c)). The third one, called supplement matrix, is used to characterize the missing or trend connections (Figure 1(d)). With the resulting matrices, we can perform community detection on the clustering matrix, and, conduct link prediction with the supplement matrix.

CLMC has several attractive benefits, most importantly:

- **Reciprocal Learning.** Instead of treat community detection and link prediction as two separated problems, CLMC learns an ideal similarity matrix to perform graph clustering and a supplement matrix to conduct link prediction simultaneously. As a result, CLMC allows finding high-quality communities since the noisy connections are well filtered and the missing connections are supplemented. In return, the resulting good graph clustering further boosts the link prediction performance.

- **Robustness:** Since CLMC examines clustering part, noise part and missing part simultaneously, it is thus quite insensitive to noisy or missing edges.
2 Proposed Approach

2.1 Low-rank Matrix Completion

Given an observed matrix $A$, if the number of known entries in $A$ is sufficiently large (i.e., $p \geq C_0 r n^{b/2} \log n$) and the entries are uniformly distributed, we can complete the data matrix by solving the following optimization problem [Candès and Recht, 2009].

$$\min \text{ rank}(\hat{A})$$
$$\text{s.t. } A_{ij} = \hat{A}_{ij} \ \forall (i,j) \in \Omega,$$

where $\hat{A}$ is the completed matrix, rank($\hat{A}$) is the rank of matrix $\hat{A}$, and $\Omega$ is the set of indices of observed entries. Although the optimization problem is NP-hard, it can be approximated to optimize the nuclear norm [Candès and Recht, 2009; Keshavan et al., 2010] as follows.

$$\min \|\hat{A}\|_*$$
$$\text{s.t. } A_{ij} = \hat{A}_{ij} \ \forall (i,j) \in \Omega,$$

where $\| \cdot \|_*$ denotes the nuclear norm of a matrix.

2.2 Cluster-driven Low-rank Matrix Completion

As aforementioned, the key idea of our algorithm is to learn an ideal clustering matrix by removing noisy edges and completing missing edges in each community. To this end, the observed data matrix $A$ can be divided into three parts: clustering matrix, noise matrix and supplement matrix. Here, we borrow the idea from low-rank matrix completion to handle missing edges. However, since the low-rank assumption of the recovered matrix $\hat{A}$ may not hold in real-world applications, unlike existing approaches, we impose the low-rank constraint on the clustering matrix instead. As a result, we first formulate our objective function as follows.

$$\min \|\hat{C}\|_*$$
$$\text{s.t. } A = \hat{C} + \hat{R} + E, \ \pi_\Omega(E) = 0, \hat{C} \geq 0, \hat{R} \geq 0$$

Where $\hat{C}, \hat{R}$ and $E$ are the clustering matrix, noise matrix and supplement matrix, respectively. $\pi_\Omega : R^{m \times n} \rightarrow R^{m \times n}$ is a linear operator that keeps the entries in $\Omega$ unchanged and sets those outside $\Omega$ zeroes.

Since we expect $\hat{C}$ to be an ideal block-diagonal matrix where the inter-cluster edges are removed, and meanwhile, the missing or trend connections are supplemented in each community, a cluster structure constraint needs to be added to make $\hat{C}$ have exactly $k$ connected components, where $k$ is the number of clusters. Let $L_{\hat{C}} = D - \hat{C}$ be the Laplacian matrix associated with $\hat{C}$, and $D$ is the diagonal matrix where the $ii$-th diagonal element is $\sum_j \hat{C}_{ij}$. If $\hat{C}$ is non-negative, according to the property of Laplacian matrix, we have the following theorem [Mohar et al., 1991; Chung, 1997].

**Theorem 1** The number $k$ of the eigenvalue 0 of the Laplacian matrix $L_{\hat{C}}$ is equal to the number of connected components in the graph associated with $\hat{C}$.

According to Theorem 1, the ideal clustering matrix can be obtained by imposing $\text{rank}(L_{\hat{C}}) = n - k$ constraint. Suppose that $\lambda_i$ is the $i$-th smallest eigenvalue of $L_{\hat{C}}$, the constraint can be approximated to the following problem for a large positive value of $\alpha_1$.

$$\min \alpha_1 \sum_{i=1}^k \lambda_i$$

When $\alpha_1$ is large enough, $\sum_{i=1}^k \lambda_i$ will be driven to be close to zero, which results in $\text{rank}(L_{\hat{C}}) = n - k$. In addition, according to the Ky Fan’s Theorem [Fan, 1949], we have

$$\sum_{i=1}^k \lambda_i = \min_{F \in R^{n \times k}, F^TF = I} \text{tr}(F^TL_{\hat{C}}F)$$

By adding the clustering constraint, our problem is further written as follows.

$$\min \|\hat{C}\|_* + \alpha_1 \text{tr}(F^TL_{\hat{C}}F)$$
$$\text{s.t. } A = \hat{C} + \hat{R} + E, \ \pi_\Omega(E) = 0, \hat{C} \geq 0, \hat{R} \geq 0$$

As for noise matrix, we assume that it is sparse and the $\ell_1$-norm constraint is imposed. As a result, we have our final objective function as follows.

$$\min \|\hat{C}\|_* + \alpha_1 \text{tr}(F^TL_{\hat{C}}F) + \alpha_2 \|\hat{R}\|_1$$
$$\text{s.t. } A = \hat{C} + \hat{R} + E, \ \pi_\Omega(E) = 0, \hat{C} \geq 0, \hat{R} \geq 0, F^TF = I$$
2.3 Optimization
In the following, we will introduce an iterative strategy to solve our optimization problem.

**Update F** By fixing $\hat{C}$, $\hat{R}$ and $E$, the problem (7) is equivalent to optimizing the following objective function:

$$\min_{F} \text{tr}(F^T L_C F) \quad \text{s.t.} \quad F \in \mathbb{R}^{n \times k}, F^T F = I \tag{8}$$

The optimal solution $F$ to the problem is formed by the $k$ eigenvectors of $L_C$ corresponding to the $k$ smallest eigenvalues.

**Update $\hat{C}$, $\hat{R}$ and $E$** By fixing $F$, we can obtain $\hat{C}$, $\hat{R}$ and $E$ by optimizing the following objective function.

$$\min \limits_{\hat{C}, \hat{R}, E} ||\hat{C}||_* + \alpha_1 \text{tr}(F^T L_C F) + \alpha_2 ||\hat{R}||_1 \tag{9}$$

s.t. $A = \hat{C} + \hat{R} + E$, $\pi_{10}(E) = 0, \hat{C} \geq 0, \hat{R} \geq 0, Z \geq 0$

Generally, it is a non-trivial task to obtain $\hat{C}$ directly. Therefore, we introduce an auxiliary variable $Z$, and optimize the following objective function.

$$\min \limits_{\hat{C}} ||\hat{C}||_* + \alpha_1 \text{tr}(F^T L_Z F) + \alpha_2 ||\hat{R}||_1 \tag{10}$$

s.t. $Z = \hat{C}, A = \hat{C} + \hat{R} + E, \pi_{10}(E) = 0, \hat{C} \geq 0, \hat{R} \geq 0, Z \geq 0$

where $L_Z$ is the Laplacian matrix of $Z$, $L_Z = D_Z - Z$.

By fixing the others, we can update $\hat{C}$ as follows. First, we introduce the partial augmented Lagrange multipliers and incorporate the equality constraints into the objective function, the problem is transformed as follows.

$$\min \limits_{\hat{C}} ||\hat{C}||_* + \alpha_1 \text{tr}(F^T L F) + \alpha_2 ||\hat{R}||_1 + \text{tr}(Y_1^T (Z - \hat{C}))$$

$$+ \frac{\mu}{2} ||Z - \hat{C}||_F^2 + \text{tr}(Y_2^T (A - \hat{C} - \hat{R} - E)) + \frac{\mu}{2} ||A - \hat{C} - \hat{R} - E||_F^2 \tag{11}$$

By fixing the others, we can update $Z$ by optimizing the following objective function.

$$\min \limits_{Z} \alpha_1 \text{tr}(F^T L_Z F) + \text{tr}(Y_1^T (Z - \hat{C})) + \frac{\mu}{2} ||Z - \hat{C}||_F^2 \tag{12}$$

Since Eq.(12) is convex and smooth, the closed-form of $Z$ can be derived as follows.

$$Z = \hat{C} - \frac{\alpha_1 H + Y_1}{\mu} \tag{13}$$

where $f_i$ is the $i$-th row of $F \in \mathbb{R}^{n \times k}$, and $H_{ij} = ||f_i - f_j||_F^2$.

Since $Z$ is non-negative, during the whole optimization process, $Z$ keeps to be non-negative by $Z = Z_+$, where $Z_+$ is the positive part of $Z$, namely, $Z_+ = \max(0, Z)$. This is the same to $\hat{C}$ and $\hat{R}$.

By fixing the others, we can update $\hat{C}$ by optimizing the following objective function.

$$\min \limits_{\hat{C}} ||\hat{C}||_* + \text{tr}(Y_1^T (Z - \hat{C})) + \frac{\mu}{2} ||Z - \hat{C}||_F^2 \tag{14}$$

**Algorithm 1 Solving Problem (7)**

1: **Input**: Data matrix $A$, parameters: $\alpha_1, \alpha_2$
2: **Output**: $\hat{C}, \hat{R}, E, F$
3: **Initialize** $Z = \hat{C} = \hat{R} = 0$, $Y_1^0 = 0, Y_2^0 = 0$, $\mu = 10^{-6}$, $max_{\mu} = 10^{10}$, $\rho = 1.5, \epsilon = 10^{-8}$
4: **while** not converge **do**
5: Fix the others and update $F$ by solving problem (8), $F = \arg \min_{F} \text{tr}(F^T L_C F)$
6: Fix the others and update $Z$ by solving problem (12), $Z = \hat{C} - (H + Y_1)/\mu$
7: Fix the others and update $\hat{C}$ by solving problem (14), $\hat{C} = D_Z \frac{1}{2\mu} (Y_1 + Y_2) + \frac{1}{2} (A + Z - \hat{R})$
8: Fix the others and update $\hat{R}$ by solving the problem (17), $\hat{R} = S_Z \frac{2}{\mu} (Y_2/\mu + (A - \hat{C}))$
9: Fix the others and update $E$ by solving problem (20), $E = \pi_{10}(A - \hat{C} - \hat{R} + \mu^{-1} Y_2)$
10: Update the Lagrange multipliers $Y_1 = Y_1 + \mu (Z - \hat{C})$
11: $Y_2 = Y_2 + \pi_{10} (\mu (A - \hat{C} - \hat{R} - E))$
12: Update the parameter $\mu$ by $\mu = \min(\mu, \mu_{max})$
13: Check the convergence conditions
14: $||A - \hat{C} - \hat{R} - E||_F^2 < \epsilon$ and $||Z - \hat{C}||_F^2 < \epsilon$
15: **end while**

By further reformulating the equation (14), the problem is equivalent to optimizing the following well-known objective function [Cai et al., 2010].

$$\min \limits_{\hat{C}} \frac{1}{2\mu} ||\hat{C}||_* + \frac{1}{2} ||\hat{C} - \frac{1}{2\mu} (Y_1 + Y_2) + \frac{1}{2} (A + Z - \hat{R} - E)||_F^2 \tag{15}$$

According to the singular value thresholding (SVT) algorithm [Cai et al., 2010], we can obtain $\hat{C}$ as follows.

$$\hat{C} = D_Z \frac{1}{2\mu} (Y_1 + Y_2) + \frac{1}{2} (A + Z - \hat{R} - E) \tag{16}$$

where the soft-thresholding operator $D_\mu$ is defined as $D_\mu(X) := U D_\mu(\Sigma) V^T, D_\mu(\Sigma) = \text{diag}(\delta_i - \mu)_+$. By fixing the others, we can update $\hat{R}$ by optimizing the following objective function.

$$\min \limits_{\hat{R}} \frac{1}{2\mu} ||\hat{R}||_1 + \frac{1}{2} ||\hat{R} - (Y_1/\mu + (A - \hat{C} - E))||_F^2 \tag{17}$$

The problem is equivalent to optimizing the following well-known objective function.

$$\min \limits_{\hat{R}} \frac{\alpha_2}{\mu} ||\hat{R}||_1 + \frac{1}{2} ||\hat{R} - (Y_2/\mu + (A - \hat{C} - E))||_F^2 \tag{18}$$

According to the shrinkage operator [Hale et al., 2007], we can obtain the $\hat{R}$ as follows.

3384
Algorithm 2 Clustering and Link Prediction via CLMC

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:</td>
<td><strong>Input</strong>: Data matrix $A$, $\alpha_1$, $\alpha_2$, number of clusters $k$</td>
</tr>
<tr>
<td>2:</td>
<td>Obtain four matrices, $\hat{C}$, $\hat{R}$, $E$ and $F$ by solving Problem (7) via Algorithm 1.</td>
</tr>
<tr>
<td>3:</td>
<td>Use $k$-means to cluster the affinity matrix $F$ into $k$ groups.</td>
</tr>
<tr>
<td>4:</td>
<td>Construct the recovered data matrix $\hat{A} = A - \frac{E + E^T}{2}$.</td>
</tr>
<tr>
<td>5:</td>
<td>Perform link prediction with $\hat{A}$.</td>
</tr>
</tbody>
</table>

\[
\hat{R} = S_{232} \left( Y_2/\mu + (A - \hat{C} - E) \right)
\]

where the shrinkage operator $S_\mu$ is defined as $S_\mu(X) := X - \mu \text{sgn}(X) + \text{sgn}(X) \odot [\mu - |X|]_+$. By fixing the others, we can update $E$ by optimizing the following objective function.

\[
\min_E \text{tr}(Y_2^T (A - \hat{C} - \hat{R} - E)) + \frac{\mu}{2} ||A - \hat{C} - \hat{R} - E||_F^2 \tag{20}
\]

To solve the problem, we can update $E$ as follows.

\[
E = \pi_0 (A - \hat{C} - \hat{R} + Y_2/\mu) \tag{21}
\]

Furthermore, the multipliers $Y_1$ and $Y_2$ are updated directly by

\[
Y_1 = Y_1 + \mu (Z - \hat{C}) \tag{22}
\]

\[
Y_2 = Y_2 + \pi_0 \left( \mu (A - \hat{C} - \hat{R} - E) \right) \tag{23}
\]

Finally, Algorithm 1 gives the pseudocode of solving Problem (7).

2.4 Graph Clustering and Link Prediction

After solving Problem (7), we can derive three matrices, including a clustering matrix $\hat{C}$, a noise matrix $\hat{R}$ and a completion matrix $E$, plus the clustering indicator matrix $F$. We utilize $F$ as a similarity matrix, and then use the $k$-means algorithm to produce the final clustering results. Moreover, since $E$ is a completion matrix, and we can use it to perform link prediction. Specifically, we can compute the completed matrix $\hat{A} = A - \frac{E + E^T}{2}$. It is important to note that community detection and link prediction work on $\hat{F}$ and $\hat{A}$, respectively, instead of the original data matrix $A$. Beyond, the two mining tasks are also mutually enhanced. On the one hand, with supplementing missing and/or trend edges while removing noisy edges, it makes clustering be more easy and effective. On the other hand, a good clustering will result in a better matrix completion to support link prediction. Finally, Algorithm 2 gives the pseudocode of graph clustering and link prediction via CLMC.

3 Experiment

We use ten publicly available real-world datasets to evaluate the performance of CLMC, including Karate, Football, Polbooks, Politics-ie, Olympics, Twitter, USAir, Celebans, PB and NS.
ranges from 15 to 40. For Spectral, Metis, FUSE and Modularity, they cannot produce promising results.

### 3.2 Community Detection Evaluation

Here, we compare CLMC to eight representatives of community detection algorithms: Ncut, Modularity, Metis, MCL, Spectral, PIC, FUSE and Attractor. **Ncut** [Shi and Malik, 2000] is a well-known algorithm for graph clustering by optimizing the *normalized cut criterion*. **Modularity** [Newman, 2006] is a popular community detection algorithm based on the *modularity* measure. **Metis** [Karypis and Kumar, 1998] is a typical clustering approach for large networks via multi-level partitioning and parallelized implementation. **MCL** [Van Dongen, 2000] is an algorithm widely used in life sciences based on the simulation of (stochastic) flow in graphs. **Spectral** is a typical clustering algorithm which allows finding arbitrarily shaped clusters [Lin and Cohen, 2010]. **PIC** is a power-iteration-based clustering approach based on spectral clustering [Ng et al., 2002]. **FUSE** is a new spectral clustering approach by employing ensemble learning and ICA [Ye et al., 2016]. **Attractor** is a model-based community detection approach based on distance dynamics [Shao et al., 2015].

For all experiments, Ncut, Modularity, Metis, Spectral, PIC and FUSE specify the number of clusters ($k = |C|$), where $|C|$ is the true number of classes. For MCL, we perform parameter tuning on the range (1.0, 4.0) with step size of 0.2. For Attractor, the cohesion parameter varies from 0.1 to 1.0 with stepsize of 0.1. For CLMC, $\alpha_1$ takes the values from (0.001, 0.1, 1) and $\alpha_2$ takes the values from (0.1, 1, 10). For all algorithms, the best results are recorded. In addition, since the performance of Spectral, PIC, FUSE and CLMC depend on the initialization with $k$-means clustering, we run them for 10 times, and the best results are reported. For networks whose communities are already known, the performance is measured by three widely used evaluation measures: Normalized Mutual Information (NMI) [Strehl and Ghosh, 2003], Adjusted Rand Index (ARI) [Rand, 1971] and Cluster Purity.

Table 1 summarizes the performance of different community detection algorithms on six real-world data sets. From the table, we can observe that CLMC achieves good performance on all data sets. On the Karate data, it even produces perfect clustering. The success of CLMC on community detection lies in the removal of noisy edges and the completion of missing/trend connections. Table 2 further reports the performance of different algorithms on the real-world data sets with missing edges of 10%. Here NMI, ARI and Purity are reported. We use bold font to mark the best results and underline the second best.

### 3.3 Link Prediction Evaluation

As an other function, CLMC also supports link prediction with the supplement matrix. Here we further evaluate its link prediction performance.
<table>
<thead>
<tr>
<th></th>
<th>CLMC</th>
<th>CN</th>
<th>Jaccard</th>
<th>HPI</th>
<th>HDI</th>
<th>LHN</th>
<th>AA</th>
<th>RA</th>
<th>PA</th>
<th>LP</th>
<th>Katz</th>
<th>LHNII</th>
<th>ACT</th>
<th>RWR</th>
<th>SimRank</th>
<th>TS</th>
<th>MF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Football</td>
<td>0.930</td>
<td>0.835</td>
<td>0.848</td>
<td>0.846</td>
<td>0.848</td>
<td>0.850</td>
<td>0.838</td>
<td>0.838</td>
<td>0.266</td>
<td>0.853</td>
<td>0.851</td>
<td>0.870</td>
<td>0.601</td>
<td>0.875</td>
<td>0.893</td>
<td>0.619</td>
<td>0.751</td>
</tr>
<tr>
<td>Karate</td>
<td>0.918</td>
<td>0.705</td>
<td>0.617</td>
<td>0.717</td>
<td>0.606</td>
<td>0.604</td>
<td>0.744</td>
<td>0.754</td>
<td>0.720</td>
<td>0.721</td>
<td>0.714</td>
<td>0.492</td>
<td>0.670</td>
<td>0.902</td>
<td>0.652</td>
<td>0.749</td>
<td>0.729</td>
</tr>
<tr>
<td>Polbooks</td>
<td>0.931</td>
<td>0.891</td>
<td>0.871</td>
<td>0.888</td>
<td>0.859</td>
<td>0.836</td>
<td>0.900</td>
<td>0.901</td>
<td>0.703</td>
<td>0.912</td>
<td>0.911</td>
<td>0.821</td>
<td>0.748</td>
<td>0.925</td>
<td>0.871</td>
<td>0.256</td>
<td>0.832</td>
</tr>
<tr>
<td>Politics-ie</td>
<td>0.960</td>
<td>0.943</td>
<td>0.940</td>
<td>0.936</td>
<td>0.932</td>
<td>0.898</td>
<td>0.947</td>
<td>0.951</td>
<td>0.721</td>
<td>0.940</td>
<td>0.940</td>
<td>0.836</td>
<td>0.749</td>
<td>0.952</td>
<td>0.904</td>
<td>0.382</td>
<td>0.921</td>
</tr>
<tr>
<td>Olympics</td>
<td>0.936</td>
<td>0.907</td>
<td>0.906</td>
<td>0.880</td>
<td>0.892</td>
<td>0.794</td>
<td>0.913</td>
<td>0.924</td>
<td>0.798</td>
<td>0.905</td>
<td>0.904</td>
<td>0.681</td>
<td>0.802</td>
<td>0.920</td>
<td>0.807</td>
<td>0.495</td>
<td>0.923</td>
</tr>
<tr>
<td>Twitter</td>
<td>0.917</td>
<td>0.896</td>
<td>0.889</td>
<td>0.893</td>
<td>0.874</td>
<td>0.828</td>
<td>0.907</td>
<td>0.913</td>
<td>0.702</td>
<td>0.904</td>
<td>0.904</td>
<td>0.741</td>
<td>0.711</td>
<td>0.910</td>
<td>0.842</td>
<td>0.494</td>
<td>0.804</td>
</tr>
<tr>
<td>USAir</td>
<td>0.965</td>
<td>0.929</td>
<td>0.893</td>
<td>0.867</td>
<td>0.887</td>
<td>0.767</td>
<td>0.940</td>
<td>0.946</td>
<td>0.877</td>
<td>0.925</td>
<td>0.922</td>
<td>0.619</td>
<td>0.894</td>
<td>0.943</td>
<td>0.782</td>
<td>0.594</td>
<td>0.921</td>
</tr>
<tr>
<td>Clegans</td>
<td>0.909</td>
<td>0.851</td>
<td>0.798</td>
<td>0.810</td>
<td>0.787</td>
<td>0.732</td>
<td>0.869</td>
<td>0.873</td>
<td>0.750</td>
<td>0.854</td>
<td>0.853</td>
<td>0.602</td>
<td>0.742</td>
<td>0.901</td>
<td>0.764</td>
<td>0.447</td>
<td>0.829</td>
</tr>
<tr>
<td>PB</td>
<td>0.943</td>
<td>0.918</td>
<td>0.873</td>
<td>0.852</td>
<td>0.869</td>
<td>0.762</td>
<td>0.921</td>
<td>0.922</td>
<td>0.901</td>
<td>0.929</td>
<td>0.929</td>
<td>0.639</td>
<td>0.884</td>
<td>0.934</td>
<td>0.774</td>
<td>0.492</td>
<td>0.928</td>
</tr>
<tr>
<td>NS</td>
<td>0.971</td>
<td>0.932</td>
<td>0.932</td>
<td>0.932</td>
<td>0.932</td>
<td>0.931</td>
<td>0.932</td>
<td>0.932</td>
<td>0.673</td>
<td>0.937</td>
<td>0.934</td>
<td>0.966</td>
<td>0.596</td>
<td>0.932</td>
<td>0.931</td>
<td>0.862</td>
<td>0.637</td>
</tr>
</tbody>
</table>

Table 3: The performance of seventeen link prediction algorithms on ten data sets. Here the AUC values are reported, and we use bold font to mark the best results and underline the second best.

Here, we compare CLMC with sixteen link prediction algorithms, including neighbor-based methods: common neighbors (CN), Jaccard, Hub Promoted Index (HPI), Hub Depressed Index (HDI), Leicht-Holme-Newman Index (LHN1 and LHN2), Adamic-Adar (AA), resource allocation (RA), preferential attachment (PA), path-based methods: Katz, local path index, average commute time (ACT), random walk with restart (RWR), transfer similarity (TS), SimRank, and latent-based method: matrix factorization (MF). For details, please refer to the link prediction survey paper [Martinez et al., 2017]. For Katz, the damping factor is searched from 0.001 and 0.01. The free parameter $\phi$ is searched in $\{0.90, 0.95, 0.99\}$. For RWR, the probability parameter $c$ is searched from 0.85 and 0.95. For MF, the number of latent factors is searched in $\{5, 10, 15, 20, 50\}$. The best results on all data sets are recorded. Moreover, the standard metric: the area under the ROC curve (AUC) [Hanley and McNeil, 1982], is used to measure the link prediction performance.

For all data sets, existing links are randomly split into: a training set (90%) and a test set (10%). After that, all link prediction approaches perform experiments on training test, and validate its performance on test set with AUC. Table 3 summarizes the link prediction performances of different algorithms on ten data sets with 10% missing connections. From Table 3, we can observe that CLMC shows its superiority over all comparing link prediction algorithms. The potential reason is that the community detection and link prediction are mutually enhanced. The clustering constraint will make more intra-connections to be supplemented.

3.4 Parameter Analysis

For CLMC, it involves two parameters $\alpha_1$ and $\alpha_2$. Figure 4 shows the different community detection performance by varying the values of $\alpha_1$ and $\alpha_2$. From the figure, we can see that CLMC can achieve a relatively stable good performance (indicating by the yellow blocks) when $\alpha_1$ ranges from 0.001 to 1, and $\alpha_2$ ranges from 0.01 to 10 on the four data sets.

4 Conclusions

In this paper, we consider community detection and link prediction simultaneously in a unified framework with cluster-driven low-rank matrix completion. Instead of working on the original network directly, the proposed framework learns a new ideal clustering matrix to perform graph clustering and a supplement matrix to conduct link prediction. Experimental results show that the proposed CLMC achieves better results on both tasks of community detection and link prediction over many state-of-the-art approaches.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61403062, 61433014, 41601025), Science-Technology Foundation for Young Scientist of Sichuan Province (2016JQ0007), Fok Ying-Tong Education Foundation for Young Teachers in the Higher Education Institutions of China (161062) and National key research and development program (2016YFB0502300).

References


