Social Network Analysis and Mining (2018) 8:43
https://doi.org/10.1007/s13278-018-0521-2

ORIGINAL ARTICLE

Overlapping community detection with preference and locality information: a non-negative matrix factorization approach

Hongyi Zhang[1] · **Xingyu Niu**[2] · **Irwin King**[3] · **Michael R. Lyu**[3]

Received: 1 March 2018 / Revised: 31 May 2018 / Accepted: 5 June 2018 / Published online: 20 June 2018

Abstract
Community detection plays an important role in understanding structures and patterns in complex networks. In real-world networks, a node in most cases belongs to multiple communities, which makes communities overlap with each other. One popular technique to cope with overlapping community detection is matrix factorization (MF). However, existing MF approaches only make use of the existence of a link, but ignore the implicit preference information inside it. In this paper, we first propose a *Preference-based Non-negative Matrix Factorization* (*PNMF*) model to take link preference information into consideration. Distinguished from traditional value approximation-based matrix factorization approaches, our model maximizes the likelihood of the preference order for each node so that it overcomes the indiscriminate penalty problem in which non-linked pairs inside one community are equally penalized in objective functions as those across two communities. Moreover, we propose a *Locality-based Non-negative Matrix Factorization* (*LNMF*) model to further incorporate the concept of locality and generalize the preference system of PNMF. Particularly, we define a subgraph called "*K*-degree local network" to set a boundary between local non-neighbors and other non-neighbors, and explicitly treat these two classes of non-neighbors in objective function. Through experiments on various benchmark networks, we show that our PNMF model outperforms state-of-the-art baselines, and the generalized LNMF model further performs better than the PNMF model on datasets with high locality.

Keywords Complex network · Community detection · Matrix factorization

1 Introduction

Unraveling community structure in a complex network has been extensively investigated in the past two decades (Fortunato and Hric 2016; Fortunato 2010). A community is intuitively regarded as a group of node closely connected inside the group but rarely making connections with nodes outside the group (Girvan and Newman 2002). In real world, communities can appear in many forms, for example, social circles manually categorized by users in ego networks (McAuley and Leskovec 2012), authors from the same institution in collaboration networks (Newman 2001), proteins with the same functionality in biochemical networks (Girvan and Newman 2002), etc. The research topic of finding such groups is known as *community detection*.

While classic community detection has a restriction that a node belongs to one and only one community, a real-world network does not usually appear this way. For example, a person in a social network is common to have multiple social identities, e.g., an alumna, a family member, a club member, a star fan, an employer, etc. Each social identity can be defined by a social community, which indicates that these communities overlap with each other. *Overlapping community detection* is the emerging topic to discover such overlapping communities, which has become the research trend in the last decade.

✉ Hongyi Zhang
zhang.hongyi@microsoft.com

Xingyu Niu
neoxyniu@gmail.com

Irwin King
king@cse.cuhk.edu.hk

Michael R. Lyu
lyu@cse.cuhk.edu.hk

[1] Microsoft Canada Development Centre, Vancouver, Canada

[2] Columbia University, New York, USA

[3] The Chinese University of Hong Kong, Shatin, Hong Kong

Unlike classic community detection and overlapping community detection cannot be directly turned into a traditional graph clustering (i.e., node clustering) problem. Thus, many heuristics have been proposed to deal with this task (Xie et al. 2013). *Matrix factorization (MF)*, as one of the standard framework to solve overlapping community detection, detects communities from a global view. Taking the adjacency matrix A of the given network as input, matrix factorization-based methods assign the number of communities in advance and seek out a node-community membership matrix F to match the information revealed by the input as accurate as possible. A typical way to achieve it is to use an objective function consisting of both A and F. Most of previous work (Psorakis et al. 2011; Wang et al. 2011) simply adopts a value-approximation-based objective function which approximates A entry by entry with the product of factorized matrices. However, an entry in A is a label representing whether there is a link or not between two nodes, but an entry in F is a real value representing the weight of a node in a community. The value-approximation-based object function clearly suffers from the mismatch between a label and a real value.

In order to tackle this mismatch issue, we propose a *Preference-based Non-negative Matrix Factorization (PNMF)* model (Zhang et al. 2015a). This model is established on two intuitions: (1) a node is more likely to build links with other nodes in the same community than those outside its community, and (2) a link can reflect the preferences of both sides to some extent, e.g., when Alice and Bob are friends, it is reasonable to argue that Alice prefers to connect with Bob than other strangers and vice versa. These intuitions can be regarded as one aspect of homophily in social networks (McPherson et al. 2001). Different from previous value-approximation-based objectives, our model maximizes the probability that a node's preference on neighbors is larger than its preference on non-neighbors, which avoids the mismatch between a real value to a label. Specifically, assuming that $A_{i,j} = 1$ and $A_{i,k} = 0$, value-approximation-based objectives try to make $F_i F_j^{\mathrm{T}}$ close to 1 and $F_i F_k^{\mathrm{T}}$ close to 0, while our preference-based objective tries to make $F_i F_j^{\mathrm{T}}$ to be larger than $F_i F_k^{\mathrm{T}}$ as much as possible but making no assumption about their values.

However, in the PNMF model, we simply separate nodes into two parts, i.e., neighbors and non-neighbors, while ignoring the fact that all non-neighbors are not supposed to be treated equally. Inspired by the famous saying "my friend's friend is also my friend", we propose a *Locality-based Non-negative Matrix Factorization (LNMF)* model (Zhang et al. 2015b) to refine the preference system of PNMF model by further splitting the non-neighbors into two parts, namely "local non-neighbors" and "distant non-neighbors". We define a "K-degree local network" to distinguish these two kinds of non-neighbors. We modify the objective function of PNMF to explicitly reflect our new assumptions: (1) neighbors are preferred to local non-neighbors, and (2) local non-neighbors are preferred to distant non-neighbors.

We employ projected stochastic gradient descent as our learning algorithm and provide efficient sampling strategies for both PNMF and LNMF models. To accelerate learning process, we implement a parallel paradigm to update parameters simultaneously with multiple processors. Experiments conducted on various benchmark datasets show that both models can outperform state-of-the-art overlapping community detection approaches and the LNMF model manages to detect communities with better quality than the PNMF model. The results provide a strong evidence that preference and locality help improve the performance of overlapping community detection.

To sum up, in this paper, we present the PNMF model (Zhang et al. 2015a) and its generalization, the LNMF model (Zhang et al. 2015b), in a systematic way, and design a parallel paradigm for both models to speed up parameter learning. We conduct comparison in experiments and show the improvement of this parallel paradigm over the original single process implementation.

Roadmap The rest of this paper is organized as follows. In Sect. 2, we review the formal definition of community detection problem and introduce some most related work in more detail. We present our PNMF and its generalized version GPNMF model in the scenario of overlapping community detection in Sects. 3 and 4, respectively. Our learning technique with sampling strategies and parameter tuning issues is illustrated in Sect. 5. We show our experimental results in Sect. 6, followed by the conclusions in Sect. 7.

2 Problem definition

To formally define the problem of *community detection*, we need to have a graph in the first place. We denote the graph as $G(V, E)$, where V is the node set and E is the link or edge set.

Definition 1 (*Community*) A community C is a subset of V with a certain characteristic.

According to this definition, nodes in a community share the same characteristic and thus are more likely to make connections with each other. Therefore, a community usually has stronger internal connections and weaker external connection, which is directly proposed as the definition of community in Girvan and Newman (2002).

Definition 2 (*Community detection*) Given a graph $G(V, E)$, community detection aims to find a set of communities

$\mathbb{S} = \{C_i | C_i \neq \emptyset, C_i \neq C_j, 1 \leq i,j \leq p\}$, which minimizes an objective function f, i.e.,

$$\arg\min_{\mathbb{S}} f(G, \mathbb{S}), \tag{1}$$

where p is the number of communities.

While classic community detection requires exhaustive and disjoint communities, i.e., $C_1 \bigcup \cdots \bigcup C_p = V$ and $C_i \bigcap C_j = \emptyset$ for any $i \neq j$, overlapping community detection has no such constraints. The relaxation makes it more realistic for real-world scenarios.

As we mentioned, matrix factorization is a popular class of methods to deal with the overlapping community detection problem. It sets the number of communities in advance and then assigns each node to its corresponding communities based on an optimization objective. We formally define matrix factorization approach for overlapping community detection as follows.

Definition 3 (*Overlapping community detection via matrix factorization*) Given a graph $G(V, E)$ with its adjacency matrix $A \in \{0,1\}^{n \times n}$, the objective of overlapping community detection via matrix factorization is to find a node-community membership matrix $F \in \mathbb{R}^{n \times p}$ whose entry $F_{u,c}$ represents the weight of node $u \in V$ in community $c \in C$ so that F can minimize a loss function l, i.e.,

$$\arg\min_{F} l(A, F), \tag{2}$$

where n is the number of nodes, p is the number of communities and C is the set of communities. In the end, we obtain the final set of communities C according to F.

3 Related work

3.1 Overlapping community detection

Massive efforts have been made on the research of community detection in the past two decades (Danon et al. 2005; Fortunato 2010; Lancichinetti and Fortunato 2009; Leskovec et al. 2010). Since most of classic community detection methods decline multiple memberships of a node, overlapping community detection attracts more and more attention in the past decade. Many approaches have been proposed to solve this specific problem (Xie et al. 2013). According to the key idea, we classify those approaches into local approaches and global approaches.

Local approaches employ a divide-and-conquer strategy to deal with subgraphs of relatively small size instead of the whole network. For example, *clique percolation* first finds all the k-cliques in the network and combines those sharing $k-1$ nodes until no combinations can be made (Kumpula et al. 2008; Palla et al. 2005). *Demon* applies label propagation algorithm to detect small communities on ego network of each node, i.e., a node with its neighbors and the corresponding links, and in the end merge those small communities with large overlap to be the final communities (Coscia et al. 2012). *Seed set expansion* approaches first picks seed nodes with a seeding strategy and runs particular algorithms on each seed node to expand the communities around it (Li et al. 2018; Whang et al. 2016).

Global approaches, on the other side, assume that communities exist in the first place and aim to assign each node to a subset of all communities. Some typical frameworks include game theory (Chen et al. 2010), stochastic block models (Airoldi et al. 2008; Jin et al. 2015), and matrix factorization (MF). Here we would like to review several important work in the matrix factorization class. Psorakis et al. (2011) is the earliest method which uses the basic $||A - WH^{\mathrm{T}}||$ as its optimization objective. Due to the vague social meaning of W and H, Wang et al. (2011) refines the objective function to $||A - FF^{\mathrm{T}}||$. Zhang and Yeung (2012) extends the matrix factorization model to matrix tri-factorization model by incorporating a community interaction matrix B, which results in a objective function of $||A - FBF^{\mathrm{T}}||$. Yang and Leskovec (2013) explicitly defines the probability of having an edge between u and v by a function of F_u and F_v, then generates the likelihood function by fitting the original graph. Most recently, Wu et al. (2018) introduce hypergraph as regularization where hyperedges are created based on structural similarity. Though these objective functions are different, they are all based on value-approximation, which is problematic because the 0 / 1 value in adjacency matrix is more like a label than a value.

3.2 Bayesian personalized ranking

Baysian personalized ranking (BPR) Rendle et al. (2009b) is originally proposed to rank items for a user in recommender systems while only implicit feedback is available. Distinguished from explicit feedback like ratings, implicit feedback only consists whether a user has labeled an item, e.g., click history, purchase history, etc. The basic assumption of BPR is that a user prefers labeled items than unlabeled ones. While traditional methods replace missing values with zeros or negative ones, BPR uses pairwise preference as training data to learn the model parameters. Technically, the objective function maximizes a posterior probability $p(\Theta| >_u)$ where Θ is a parameter and $>_u$ is a latent preference structure for user u. BPR has become a widely-used model in one-class collaborative filtering and there are several further work on top of it. For example, Rendle et al. (2009a) extend the original matrix factorization to a tensor factorization to

recommend personalized tags for a user given an item. Zhao et al. (2014) leverage social connections to improve item recommendations by building a new preference system.

We adopt the basic assumption of BPR into the overlapping community detection task. To be specific, a node prefers its neighbors and its "non-neighbors". Our objective function is to maximize the probability $p(>_u |F)$, where F is a nonnegative matrix representing the latent node-community membership. Moreover, our extension of the basic assumption by incorporating locality is similar to that in Zhao et al. (2014).

4 A preference-based non-negative matrix factorization (PNMF) model

In this section, we introduce our PNMF model for overlapping community detection. We start with model assumptions, followed by model formulation and parameter learning scheme. Particularly, we will briefly illustrate our parallel paradigm for parameter learning acceleration.

4.1 Preliminaries

The set of i's neighbors is denoted by $N^+(i)$ and $N^-(i) := N^+(i)^c \backslash \{i\}$ is defined as the "non-neighbors" of i, where $N^+(i)^c$ denotes the complement set of $N^+(i)$. It can be inferred that $V = N^+(i) \cup N^-(i) \cup \{i\}$ for every i. Moreover, we define a learning set $S : V \times V \times V$ by

$$S = \{(i,j,k) | i \in V, j \in N^+(i), k \in N^-(i)\},$$

which consists of all the triples (i, j, k), where j is a neighbor of i while k is not.

4.2 Model assumption

Our PNMF model is based on the intuitive idea that two nodes are more likely to connect with each other if they share more common communities. If we treat a link as the consequence of mutual preference, we can safely claim that a node has a higher preference on its neighbors than non-neighbors. Thus, the main assumption of our PNMF model can be formally denoted as

$$r_{u,i} \geq r_{u,j}, \quad \text{if } A_{u,i} = 1 \text{ and } A_{u,j} = 0, \tag{3}$$

where $r_{u,i}$ is the preference of node u on node i, i.e., how much u wants to build a link with i, and $A_{u,i}$ is the corresponding entry in adjacency matrix A.

Moreover, we need two more independent assumptions before we start model formulation.

1. *Node independence*. The preference order of each node is independent with that of any other node. There will be a link between u and v if and only if u prefers to connect with v and symmetrically v prefers to connect with u.
2. *Pair independence*. For a fixed node i, its preference between pair (j, k) is independent with its preference between pair (u, v) given $(j,k) \neq (u,v)$, $j, u \in N^+(i)$ and $k, v \in N^-(i)$.

4.3 Model formulation

The goal of our PNMF model is to find a node-community membership matrix that maximizes the likelihood of observed preference order for all the nodes. According to the "node independence" assumption, the overall likelihood can be denoted as a product of likelihood of each node. Thus, our objective function can be written as

$$\max_{F \in \mathbb{R}_+^{n \times p}} \prod_{i \in V} \mathcal{P}(>_i |F), \tag{4}$$

where $>_i$ denotes the observed preferences for node i according to adjacency matrix A and F is the node-community membership matrix.

According to our main assumption as well as the "pair independence" assumption, the probability of preference order for a single node i can be written as

$$\begin{aligned} \mathcal{P}(>_i |F) &= \textstyle\prod_{(j,k) \in V \times V} \mathcal{P}(j >_i k|F)^{\delta(j \in N^+(i)) \delta(k \in N^-(i))} \\ &\quad \cdot (1 - \mathcal{P}(j >_i k|F))^{1 - \delta(j \in N^+(i)) \delta(k \in N^-(i))} \\ &= \textstyle\prod_{(j,k) \in V \times V} \mathcal{P}(j >_i k|F)^{\delta((i,j,k) \in S)} \\ &\quad \cdot (1 - \mathcal{P}(j >_i k|F))^{\delta((i,j,k) \notin S)}, \end{aligned} \tag{5}$$

where S is the learning set mentioned in preliminaries and δ is the indicator function, i.e.,

$$\delta(a) = \begin{cases} 1 & \text{if } a \text{ is true,} \\ 0 & \text{else.} \end{cases}$$

For a triple (i, j, k), if $(i,j,k) \in S$, then $(i,k,j) \notin S$. Given $\mathcal{P}(j >_i k|F) + \mathcal{P}(k >_i j|F) = 1$, it is obvious that $\mathcal{P}(j >_i k|F)^{\delta((i,j,k) \in S)} = (1 - \mathcal{P}(k >_i j|F)^{\delta((i,k,j) \notin S)}$. Applying this to Eq. (5), maximizing $\mathcal{P}(>_i |F)$ is equivalent to

$$\max_{F \in \mathbb{R}_+^{n \times p}} \prod_{(j,k) \in V \times V} \mathcal{P}(j >_i k|F)^{\delta((i,j,k) \in S)}. \tag{6}$$

Combining Eqs. (4) and (6), our objective function can be rewritten as

$$\max_{F \in \mathbb{R}_+^{n \times p}} \prod_{(i,j,k) \in S} \mathcal{P}(j >_i k|F). \tag{7}$$

Now the problem comes to how to define the preference of one node on another. A node's community membership vector is represented by the corresponding row of the node-community membership matrix F. Thus, more community two nodes share, more similar their membership vectors

will be. Considering both our intuition that two nodes have a higher probability to be linked if they share more communities and our main assumption that a node has a higher preference on its neighbors, we define the probability that i prefers j than k given the node-community membership matrix as

$$\mathcal{P}(j >_i k|F) = \sigma(F_i \cdot F_j^{\mathrm{T}} - F_i \cdot F_k^{\mathrm{T}}), \tag{8}$$

where σ is the sigmoid function $\sigma(x) := \frac{1}{1+e^{-x}}$.

The sigmoid function can map any real number into (0, 1). The probability that i prefers j than k is 0.5 when $F_i F_j^{\mathrm{T}} = F_i F_k^{\mathrm{T}}$. Additionally, this probability approaches 0 when $F_i F_j^{\mathrm{T}} \ll F_i F_k^{\mathrm{T}}$ and approaches 1 when $F_i F_j^{\mathrm{T}} \gg F_i F_k^{\mathrm{T}}$. These properties precisely satisfy the requirements of our model.

For simplicity, we define $\hat{x}(i,j) := F_i \cdot F_j^{\mathrm{T}}$. Equation (8) can be rewritten as

$$\mathcal{P}(j >_i k|F) = \sigma(\hat{x}(i,j) - \hat{x}(i,k)). \tag{9}$$

Now combining Eqs. (7), (8), and (9), we can construct the final objective function of our PNMF model as

$$\begin{aligned} l(F) &:= \max_{F \in \mathbb{R}_+^{n \times p}} \ln \prod_{(i,j,k) \in S} \mathcal{P}(j >_i k|F) - \lambda \cdot reg(F) \\ &= \max_{F \in \mathbb{R}_+^{n \times p}} \sum_{(i,j,k) \in S} \ln \mathcal{P}(j >_i k|F) - \lambda \cdot reg(F) \\ &= \max_{F \in \mathbb{R}_+^{n \times p}} \sum_{(i,j,k) \in S} \ln \sigma(\hat{x}(i,j) - \hat{x}(i,k)) - \lambda \cdot reg(F), \end{aligned} \tag{10}$$

where $reg(F)$ is the regularization term we add to avoid over-fitting and λ is the regularization parameter. We choose Frobenius norm as the regularization term, i.e., $reg(F) = \frac{1}{2}||F||_F^2$, because it is differentiable and fits our parameter learning process.

4.4 Parameter learning

We employ the widely used stochastic gradient descent (SGD) as our learning method because it is efficient and easy to implement. In each updating step, SGD randomly selects a triple in the learning set S and updates the corresponding parameters Θ by walking along the gradient direction,

$$\Theta^{t+1} = \Theta^t + \alpha \frac{\partial l}{\partial \Theta}, \tag{11}$$

where α is the learning rate. Specifically, the derivative of Eq. (11) is calculated by

$$\begin{aligned} \frac{\partial l}{\partial \Theta} &= \frac{\partial}{\partial \Theta} \ln \sigma(\hat{x}(i,j) - \hat{x}(i,k)) - \lambda \frac{\partial}{\partial \Theta} reg(F) \\ &= \frac{-e^{\hat{x}(i,k)-\hat{x}(i,j)}}{1+e^{\hat{x}(i,k)-\hat{x}(i,j)}} \cdot \frac{\partial}{\partial \Theta}(\hat{x}(i,j) - \hat{x}(i,k)) - \lambda \Theta \end{aligned} \tag{12}$$

and

$$\frac{\partial}{\partial \Theta}(\hat{x}(i,j) - \hat{x}(i,k)) = \begin{cases} F_{j,t} - F_{k,t} & \text{if } \Theta = F_{i,t} \\ F_{i,t} & \text{if } \Theta = F_{j,t} \\ -F_{i,t} & \text{if } \Theta = F_{k,t} \\ 0 & \text{else} \end{cases}, \tag{13}$$

where λ is the regularization parameter. Regarding the non-negativity constraint, we exploit the idea of projected gradient methods for non-negative matrix factorization, which maps the value of a parameter back to non-negativity (Lin 2007).

The whole learning process is described in Algorithm 1 and the sampling strategy is described in Algorithm 2. Let sample size be t. The time complexity of each iteration is $O(tp)$ and the space complexity is $O(np)$, where n is the number of nodes and p is the number of communities, since we need to save the node-community membership matrix into memory.

Algorithm 1 Community detection using PNMF

Input: A, the adjacency matrix of original graph
Output: F, the node-community membership matrix
1: initialize F
2: compute initial loss
3: **repeat**
4: **for** *num_samples* $= 1$ to *sample_size* **do**
5: sample (u, i, j) according to Algorithm 2
6: **for** each entry Θ in F_u, F_i and F_j **do**
7: update Θ according to Equation (13)
8: $\Theta \leftarrow \max(\Theta, 0)$
9: **end for**
10: **end for**
11: compute loss
12: **until** convergence or *max_iter* is reached

Algorithm 2 Sampling Strategy of PNMF

Input: A, the adjacency matrix of original graph
Output: (u, i, j, d), a quadruple to perform a step in stochastic gradient descent
1: sample node u from V uniformly at random
2: sample node i from $N^+(u)$ uniformly at random
3: sample node j from $N^-(u)$ uniformly at random

4.5 A parallel paradigm for accelerating parameter learning

In Algorithm 1, the execution time for each iteration is proportional to the sample size, and it increases significantly when the network scales to hundreds of thousands of nodes. To accelerate this learning process, we design a parallel paradigm to distribute stochastic gradient descent (SGD) tasks with multiple processes.

The most important steps in Algorithm 1 in terms of time are Step 5 to Step 10, i.e., triples sampling and parameters updating. The idea of our parallel paradigm is pretty straightforward. Given n sub-processes, each sub-process conducts sampling and updating independently in one iteration. Since the node-community membership matrix F costs a lot of memory to store, we can not make a copy of it for each process. In ideal case, we expect this paradigm to run as n times fast as the single process paradigm. However, while sampling always works in parallel, the updating part suffers synchronization problem. Here we discuss two possible solutions.

The first solution is to simply use locks. When a sub-process performs updating on a particular row of the node-community membership matrix, a lock will be assigned to this row to prevent other sub-processes from updating it. The correctness of this paradigm can be easily justified since two processes only perform updating simultaneously when there is no collision. However, if collisions happen a lot, the computational efficiency will be massively affected and in the worst case, the time cost will be as much as the single process paradigm.

The other solution is a lock-free paradigm (Recht et al. 2011). In this paradigm, each sub-process has read-only access to the global node-community membership matrix and keeps the updating results on its own memory. After each iteration, the main process performs updating on the global node-community membership matrix according to the information it collects from all sub-processes. To be specific, for each row, the changes Δ from all sub-processes are summed and the sum is added to this row of the global node-community membership matrix. Thus, in one iteration, each sub-process starts from the same node-community membership matrix from the previous iteration and is totally independent with each other. When no collision happens, this lock-free paradigm is exactly the same as the previous paradigm. On the contrast, when collisions happen a lot, this paradigm prevents any waiting time among sub-processes in one iteration. Although it may introduce some computation errors in updating, it still walks along the gradient in the right direction. Since we care more about the speed of our parallel paradigm, this lock-free one is obviously a better choice for implementation.

4.6 Other issues

There are several remaining details to be discussed.

- *The number of communities.* The nature of matrix factorization needs us to set the number of communities which are unknown in advance. A cross-validation paradigm is used. In details, we reserve 10% of nodes as validation set at first. After learning the node-community membership matrix F, we compute the sum of log-likelihood function for all nodes in validation set via Eq. (17). Since the computational cost is huge for cross-validation, only a small set of quadruple will be sampled.
- *The community membership threshold.* Obtaining the node-community membership matrix F is still one step away from getting the final node-community correspondence. We need to set a threshold δ to decide whether a community accepts a node. If $F_{u,c} \geq \delta$, we say that node u belongs to community c. According to Eq. (8), we need $p(j >_i k|F)$ to be closer to 1 than 0 if i prefer j than k. We assume that i, j share exactly one community and i, k do not share any communities. Thus $F_i F_k^{\mathrm{T}} = 0$. Due to the symmetry of i and j, we have

$$\sigma(F_i F_j^{\mathrm{T}} - F_i F_k^{\mathrm{T}}) = \sigma(\delta^2 - 0) = \frac{1}{1 + e^{-\delta^2}} = \beta,$$

  where β is in the range of (0.5, 1). When β is given, we can compute δ by

$$\delta = \sqrt{-\ln\left(\frac{1}{\beta} - 1\right)}. \tag{14}$$

- *The convergence criterion.* First, we randomly generate a subset of triples to be our loss sample and compute initial loss on this set according to Eq. (10). After each iteration, we need to compute loss again and we stop stochastic gradient descent when the absolute difference between

Table 1 A summary of notations

Notation	Meaning
$G(V, E)$	Graph G with node set V and edge set E
$L_K(u)$	u's K-degree local network in G
$V_K(u)$	Node set of $L_K(u)$ except u itself
$S_K(u)$	Node set of u's K-degree local non-neighbors
$T_K(u)$	Node set of u's K-degree distant non-neighbors
$N^+(u)$	Node set of u's neighbors
$N^-(u)$	Node set of u's non-neighbors

current loss and previous loss is smaller than a very small percentage, say ϵ, of initial loss.

5 A locality-based non-negative matrix factorization (LNMF) model

In this section, we first define the concept of K-degree locality and formalize our LNMF model in the scenario of community detection. Then we discuss about our parameter learning process including sampling strategy.

5.1 Preliminaries

Definition 4 (*K-degree local network*) Given an undirected and unweighted graph G, for a node $u \in G$, u's K-degree local network $L_K(u)$ is the subgraph consisting of all nodes whose shortest path length to u is less than or equal to K.

According to the definition above, $L_0(u)$ consists of only node u, $L_1(u)$ is the subgraph including node u and all its neighbors, $L_\infty(u)$ is the whole graph, etc. We denote the node set of $L_t(u)$ except u itself as $V_t(u)$, where $t = 1, 2, ...$

Now we further define the terms of "local non-neighbors" and "distant non-neighbors".

Definition 5 (*K-degree local non-neighbors*) Given a K-degree local network $L_K(u)$, the set of K-degree local non-neighbors $S_K(u)$ is defined as $S_K(u) := L_K(u) \backslash L_1(u)$, where $K \geq 1$.

Definition 6 (*K-degree distant non-neighbors*) Given a K-degree local network $L_K(u)$, the set of K-degree distant non-neighbors $T_K(u)$ is defined as $T_K(u) := L_\infty(u) \backslash L_K(u)$, where $K \geq 1$.

A summary of notations is shown in Table 1. Four simple propositions can be drawn from the above notations.

Proposition 1 $V_K(u) = N^+(u) \bigcup S_K(u)$.

Proposition 2 $N^+(u) \bigcap S_K(u) = \emptyset$.

Proposition 3 $N^-(u) = S_K(u) \bigcup T_K(u)$.

Proposition 4 $S_K(u) \bigcap T_K(u) = \emptyset$.

5.2 Model assumption

Recall the main assumption of the PNMF model in Eq. (3). Incorporating the concept of K-degree local network, we can employ K-degree local non-neighbors to enhance the model assumption. The new model assumption for our LNMF model can be represented as

$$r_{i,j} \geq r_{i,k}, r_{i,k} \geq r_{i,d}, j \in N^+(i), k \in S_K(i), d \in T_K(i), \tag{15}$$

where $r_{i,j}$ is still the preference of node i on node j. It can be interpreted as (1) neighbors are preferred to local non-neighbors, (2) local non-neighbors are preferred to distant non-neighbors. We notice that when $K = 1$, $S_K(i) = \emptyset$ and $T_K(i) = N^-(i)$. In this case, our model assumption degrades to the one of the PNMF model. Thus, the LNMF model can be viewed as a generalization of the PNMF model.

We also adopt the two independence assumptions listed in the Sect. 3.2, i.e., *node independence* and *pair independence*, when formalizing our LNMF model.

5.3 Model formulation

Given model assumptions, we are ready to present our LNMF model in a formal way. Since nodes are independent of each other, we can consider one node at first.

For a node i, the optimization criterion is to maximize the likelihood of preference order which can be represented as a product of pairwise preferences, i.e.,

$$\begin{aligned} &\prod_{j,k \in V_K(i)} [\mathcal{P}(r_{i,j} \geq r_{i,k}|F)^{\delta(i,j,k)} \\ &\qquad (1 - \mathcal{P}(r_{i,j} \geq r_{i,k}|F))^{1-\delta(i,j,k)}] \cdot \\ &\prod_{k,d \in N^-(i)} [\mathcal{P}(r_{i,k} \geq r_{i,d}|F)^{\xi(i,k,d)} \\ &\qquad (1 - \mathcal{P}(r_{i,k} \geq r_{i,d}|F))^{1-\xi(i,k,d)}], \end{aligned} \tag{16}$$

where $\delta(\cdot)$ and $\xi(\cdot)$ are two indicator functions that

$$\delta(i, j, k) = \begin{cases} 1 & \text{if } j \in N^+(i) \text{ and } k \in S_K(i), \\ 0 & \text{otherwise} \end{cases}$$

and

$$\xi(i, k, d) = \begin{cases} 1 & \text{if } k \in S_K(i) \text{ and } d \in T_K(i), \\ 0 & \text{otherwise} \end{cases}.$$

Recall the four propositions in preliminaries that $V_K(i)$ and $N^-(i)$ can be split into two disjoint sets with different levels of preference. Following the scheme proposed in Rendle et al. (2009b) and Zhao et al. (2014), we can simplify Eq. (16) to

$$\frac{\sum_{j\in N^+(i),k\in S_K(i)} \mathcal{P}(r_{i,j}\geq r_{i,k}|F)}{|N^+(i)|\cdot|S_K(i)|}+ \frac{\sum_{k\in S_K(i),d\in T_K(i)} \mathcal{P}(r_{i,k}\geq r_{i,d}|F)}{|S_K(i)|\cdot|T_K(i)|}. \quad (17)$$

Applying the sigmoid function $\sigma(x) := \frac{1}{1+e^{-x}}$ to interpret $\mathcal{P}(r_{i,j} \geq r_{i,k}|F)$, i.e., $\mathcal{P}(r_{i,j} \geq r_{i,k}|F) = \sigma(\hat{x}(i,j) - \hat{x}(i,k))$, we sum up the log-likelihood functions of all nodes:

$$\sum_i [\ \sum_{j\in N^+(i),j\in S_K(i)} \ln \sigma(\hat{x}(i,j) - \hat{x}(i,k)) + \lambda(i) \cdot \sum_{k\in S_K(i),d\in T_K(i)} \ln \sigma(\hat{x}(i,k) - \hat{x}(i,d))], \quad (18)$$

where $\hat{x}(i,j) := F_i \cdot F_j^{\mathrm{T}}$ can be regarded as the correlation between i and j, and $\lambda(i) := \frac{|N^+(i)|}{|T_K(i)|}$ can be regarded the coefficient of locality influence.

In the end, to prevent our model from over-fitting, we add a regularization term $reg(F) = \frac{1}{2}||F||_F^2$, which is the Frobenius norm of the node-community membership matrix. Thus, the final objective function l is written as

$$l(F) = \sum_i [\sum_{j\in N^+(i),k\in S_K(i)} \ln \sigma(\hat{x}(i,j) - \hat{x}(i,k)) + \lambda(i) \cdot \sum_{k\in S_K(i),d\in T_K(i)} \ln \sigma(\hat{x}(i,k) - \hat{x}(i,d))] - \lambda_r reg(F), \quad (19)$$

where λ_r is the regularization coefficient.

5.4 Parameter learning

Parameter learning for the LNMF model is mostly the same as that for the PNMF model except that we now need to sample a set of (source, neighbor, local non-neighbor, distant non-neighbor) quadruples instead of (source, neighbor, non-neighbor) triples at each learning step. The learning process for the LNMF model is described in Algorithm 3 and the sampling strategy is described in Algorithm 4. For the sampling of k, we need to pre-process the whole graph to record a set of local nodes of each i in the graph. Based on the fact that $N^-(i) = S_K(i) \bigcup T_K(i)$, we keep sampling a random node until we get a node d neither in $N^+(i)$ nor in $S_K(i)$.

The same parallel paradigm as what we employ in the PNMF model is also adopted to accelerate the LNMF model. In each iteration, multiple sub-processes sample the quadruples and update the node-community membership matrix F stored in the main process in parallel. We expect the parallel version to run much faster than the single process version as well.

Table 2 Statistics of six Newman's datasets

Dataset	V	E
Dolphins	62	159
Les Misérables	77	254
Books about US politics	105	441
Word adjacencies	112	425
American college football	115	613
Coauthorship in network science	1589	2742

V number of nodes, E number of links

Algorithm 3 Community Detection via *LNMF*

Input: A, the adjacency matrix of original graph
Output: F, the node-community membership matrix
initialize F
compute initial loss
repeat
  for *num_samples* $= 1$ to *sample_size* **do**
    sample (i, j, k, d) according to Algorithm 4
    for each entry Θ in F_i, F_j, F_k and F_d **do**
      update Θ according to Equation (11)
      $\Theta \leftarrow \max(\Theta, 0)$
    end for
  end for
  compute loss
until convergence or *max_iter* is reached

Algorithm 4 Sampling Strategy of LNMF

Input: A, the adjacency matrix of original graph
Output: (i, j, k, d), a quadruple to perform a step in stochastic gradient descent
sample node i from V uniformly at random
sample node j from $N^+(i)$ uniformly at random
sample node k from $S_K(i)$ uniformly at random
sample node d from $T_K(i)$ uniformly at random

5.5 Discussion on further extension

Our LNMF model can be further generalized by extending the preference system from three levels to n levels. Mathematically, according to the notations in Sect. 4.1, let $P_K(u) := L_K(u) \backslash L_{K-1}(u)$ for $K = 1, 2, \ldots, n$ be the node set of a particular preference level of node u and we assume that u's preference on $P_i(u)$ is larger than its preference on $P_j(u)$ if $i > j$. For model formulation, we only need to modify a few spots to make it a more general one. However, our learning algorithm is not efficient enough. Specifically, a trivial extension of our sampling strategy suffers from two problems: (1) the upper limit of K can change for different nodes, (2) for a node u, pre-processing the whole graph to record node sets of all preference levels is equal to a breath-first search starting from u, which is too time-consuming for large-scale networks. We do not yet have an adequate solution to this problem.

6 Experiments

In this section, we compare our PNMF and LNMF models with both classic and state-of-the-art overlapping community detection approaches on both synthetic and real-world datasets. The experimental results are evaluated by three metrics, namely modularity, normalized mutual index (NMI) and F_1 score. We also compare our parallel implementation with the single process one.

6.1 Data description

For synthetic datasets, we use the Lancichinetti–Fortunato–Radicchi (LFR) benchmark networks (Lancichinetti et al. 2008). We generate three binary undirected networks of 1000 nodes with different settings.

For real-world datasets, We have six benchmark networks collected by Newman[1] which have no ground-truth communities and relatively small sizes. Simple statistics of these datasets can be found in Table 2. We also have three networks with ground-truth communities collected by SNAP[2] (Yang and Leskovec 2015):

- *Amazon*: a products co-purchasing network based on *customers who bought this item also bought feature* of the Amazon website;
- *DBLP*: a collaboration network of research paper authors in computer science;
- *YouTube*: a social network of a video-sharing website.

[1] http://www-personal.umich.edu/~mejn/netdata/.

[2] http://snap.stanford.edu/data/.

Table 3 Statistics of three SNAP datasets

Dataset	V	E	C	U
DBLP	317k	1.0M	2.5k	429.8
Amazon	335k	926k	49k	100.0
YouTube	1.1M	3.0M	30k	9.7

V number of nodes, E number of links, C number of ground-truth communities, U average number of nodes per community

These SNAP networks have large sizes so that we can use them to test the scalability of our models. The statistics are shown in Table 3.

6.2 Experimental setup

We conduct our experiments on a computer with 32 Xeon 2.40 GHz CPUs and 128 GB memory.

6.2.1 Baselines

We select both classic and state-of-the-art methods to compare with our models.

- *SCP* (Kumpula et al. 2008) accelerates the original *CP* method (Palla et al. 2005) in a sequential manner. We set k to be 4 or 5 when finding k-cliques.
- *LC* (Ahn et al. 2010) clusters link instead of node to get overlapping communities. We ignore all communities with only one or two nodes because they are meaningless.
- *Demon* (Coscia et al. 2012) uses label propagation algorithm to detect small communities on ego network of each node and then merge communities with large overlap.
- *BigCLAM* (Yang and Leskovec 2013) is also claimed as a scalable model. It can search for the best number of communities given a range.

6.2.2 Evaluation metrics

We use modularity as the evaluation metric for datasets without ground-truth communities, NMI and F_1 score for datasets with ground-truth communities.

- *Modularity*. The classic modularity is defined as

$$Q = \frac{1}{2|E|} \sum_{u,v \in V} (G_{u,v} - \frac{d(u)d(v)}{2|E|}) I_{u,v},$$

where $d(u)$ is the degree of node u, $G_{u,v}$ is the (u, v) entry of the adjacency matrix G, and $I_{u,v} = 1$ if u, v are in the same community otherwise 0 (Newman 2006). In the overlapping scenario, since a node pair may share more than one communities, a minor modification is made by

replacing $I_{u,v}$ with $|C_u \cap C_v|$, i.e., the number of overlapped community between u and v:

$$\hat{Q} = \frac{1}{2|E|}\sum_{u,v\in V}\left(G_{u,v} - \frac{d(u)d(v)}{2|E|}\right)|C_u \cap C_v|.$$

From the definition, we can see that greater value of modularity reveals denser connectivity within the detected communities because only linked node pairs sharing common communities contribute positively to the value. This metric has been frequently used in other MF-based works as well (Yang and Leskovec 2013; Zhang and Yeung 2012; Psorakis et al. 2011). As we know, modularity has been directly used as an optimization objective in community detection, and those approaches are called modularity-based methods (Clauset et al. 2004; Duch and Arenas 2005; Guimera et al. 2004; Newman 2004). However, when we compare the quality of detected communities among non-modularity-based models, modularity can still be served as a useful metric.

- *NMI*. Normalized mutual information is very popular in evaluating community detection algorithms (Lancichinetti et al. 2009). The mutual information between two community partitions A and B is defined as the Kullback–Leibler (KL) divergence between joint distribution $P_{AB}(a,b)$ and the product of two marginal distributions $P_A(a)P_B(b)$

$$I(P_A, P_B) = H(P_A) + H(P_B) - H(P_{AB}),$$

where $H(P_A) = -\sum_{a\in A} P_A(a)\log P_A(a)$ is the entropy of distribution P_A and $H(P_{AB})$ is the entropy of the joint distribution P_{AB}. Here $P_A(a) = \frac{n_a}{n}$ and $P_{AB}(a,b) = \frac{n_{ab}}{n}$, where n is number of all nodes, n_a is the number of nodes in community a, and n_{ab} is the number of nodes in both community a and b. After normalization, NMI is defined as

$$NMI(P_A, P_B) = \frac{2I(P_A, P_B)}{H(P_A) + H(P_B)}.$$

In our experiment, we employ a more conventional normalization corrected for overlapping scenario to avoid unintentional behaviors (McDaid et al. 2011).

- F_1 score. F_1 score is also one of the best measurements for datasets with ground-truth communities. The F_1 score of a detected community S_i is defined as the harmonic mean of precision(S_i) and recall(S_i), where precision(S_i) and recall(S_i) are defined as

$$\text{precision}(S_i) = \max_j \frac{C_j \bigcap S_i}{|C_j|},$$

and

$$\text{recall}(S_i) = \max_j \frac{C_j \bigcap S_i}{|S_i|},$$

where C_j is the node set of a ground-truth community. The average F_1 score for the set of detected communities S is

$$\overline{F_1}(S) = \frac{1}{|S|}\sum_{S_i\in S} F(S_i).$$

6.2.3 Setting the degree

Recall that if we set $K = 1$ in K-degree local network, the LNMF model will degrade to the PNMF model. We've tried both $K = 2$ and 3 on our datasets. The result shows that the average number of common communities two nodes in a K-degree local network share is about the same. For $K \geq 4$, generating K-degree local network costs too much time. Thus, we set $K = 2$ for all experiments, which means only a friend's friends are regarded as local non-neighbors.

6.3 Results

The experiments in Sects. 6.3.1 and 6.3.2 are conducted on a single process. The experiments in Sect. 6.3.3 are conducted on multiple processes.

6.3.1 Performance on LFR benchmark

For all synthetic networks, we set number of nodes as 1000, average degree as 10, maximum degree as 50, exponent for the degree distribution as 2, exponent for the community size distribution as 1 and mixing parameter as 0.2. The only two variants are number of overlapping nodes (on) and number of memberships of the overlapping nodes (om). We use both NMI and F_1 score as metrics and the results are shown in Tables 4 and 5, respectively. We can see that the PNMF model is comparable to other methods while the LNMF model outperforms the others.

6.3.2 Performance on real-world datasets

We set the regularization coefficient to be around 0.001 and the convergence parameter ϵ to be 0.001 for all experiments. The sample size t is determined according to data size. For UMich networks, we set $t = m$, i.e., the number of links. For SNAP networks, we set $t = 10\sqrt{n}$ to finish one iteration without taking too much time, where n is the number of nodes. The maximum times of iteration is set to 100, though in fact all datasets converge before reaching the limit.

The experimental results on UMich networks are shown in Table 6. We use modularity as the metric since these networks have no ground-truth communities. The comparison

Table 4 Experimental results on LFR benchmark datasets in terms of NMI

on	om	SCP	LC	Demon	BigCLAM	PNMF	LNMF (RI)
40	2	0.593	0.329	0.625	0.502	0.598	0.643 (7.5%)
40	4	0.369	0.318	0.456	0.534	0.464	0.547 (17.9%)
80	2	0.416	0.370	0.441	0.484	0.465	0.501 (7.7%)

RI relative improvement over PNMF

Table 5 Experimental results on LFR benchmark datasets in terms of F_1 score

on	om	SCP	LC	Demon	BigCLAM	PNMF	LNMF (RI)
40	2	0.399	0.316	0.428	0.376	0.417	0.442 (6.0%)
40	4	0.314	0.308	0.380	0.387	0.373	0.411 (10.2%)
80	2	0.329	0.326	0.378	0.364	0.361	0.394 (9.1%)

Table 6 Experimental results on UMich datasets in terms of modularity

Dataset	SCP	LC	Demon	BigCLAM	PNMF	LNMF (RI)
Dolphins	0.305	0.654	0.680	0.423	0.979	1.086 (10.9%)
Les Misérables	0.307	0.773	0.026	0.540	1.103	1.184 (7.3%)
Books about US politics	0.496	0.851	0.432	0.529	0.864	1.270 (47.0%)
Word adjacencies	0.071	0.271	0.032	0.231	0.668	0.701 (4.9%)
American College football	0.605	0.891	0.540	0.518	1.049	1.235 (17.7%)
Network science	0.729	0.956	0.642	0.503	1.657	2.310 (39.4%)

clearly reveals the dominant superiority of our PNMF and LNMF model over baseline methods, especially MF-based approaches. Moreover, the LNMF model performs slightly better than the PNMF model, which validates the usage of locality information.

The experimental results on SNAP networks are shown in Table 7. F_1 score is used as measurement because these datasets have ground-truth communities. Here we only compare with BigCLAM because all the other baselines can not scale to datasets with such sizes. This fact can reflect the scalability of our PNMF and LNMF model to some extent. From the results we can see that both of our models outperform the BigCLAM model overall. Specifically, the LNMF model has fair improvement over the PNMF model on both DBLP and Amazon datasets. For the YouTube dataset, we conduct a data observation and figure out that most of its communities are quite sparse due to the small size of communities and large variety of users. In other words, locality information is not useful on the YouTube dataset. This explains the reason why the LNMF model fails to have improvement on this dataset.

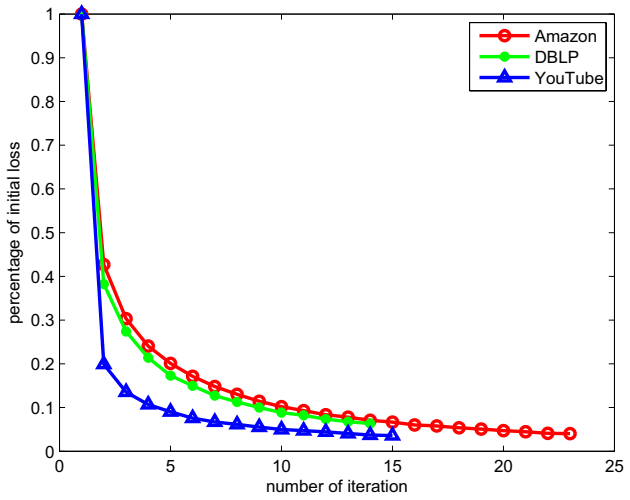
During parameter learning of both PNMF and LNMF models, the running time of one iteration is about 1–2 h for DBLP and Amazon and about 4–5 h for YouTube. Figure 1a, b demonstrates the convergence speed of the learning algorithm of the PNMF model and LNMF model on SNAP datasets, respectively. Each point in the figure represents the ratio of current loss to initial loss after a certain number of iterations. The results show that both of our models can converge within a fair number of iterations. However, we notice that the final loss of the LNMF model on the YouTube dataset is quite large compared with the other two datasets. This observation actually matches our previous discussion in Table 7 that locality is possibly not a feature of the YouTube dataset.

Table 7 Experimental results on SNAP datasets in terms of F_1 score

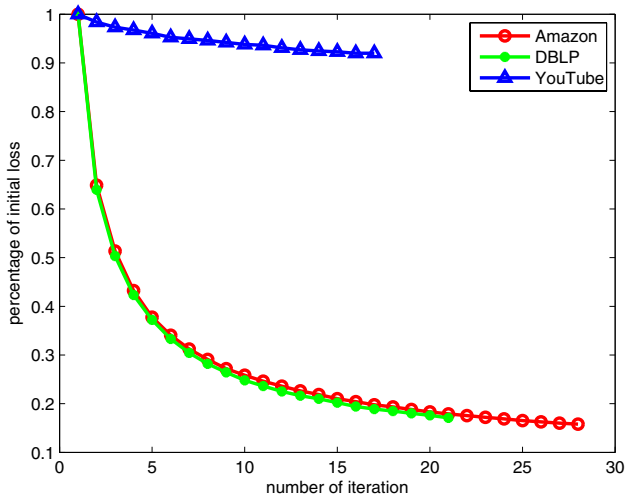
Dataset	BigCLAM	PNMF	LNMF (RI)
DBLP	0.039	0.098	0.107 (9.2%)
Amazon	0.044	0.042	0.048 (11.4%)
YouTube	0.019	0.060	0.057 (− 5.0%)

6.3.3 Performance of parallel paradigm

With the same model parameter setting as single process experiments, we test the efficiency of our parallel paradigm on SNAP datasets. Figure 2 shows the average training time the PNMF model and the LNMF model take for one iteration by using different numbers of processes, respectively. The number of total samples per iteration remains the same so that we can fairly compare the average time per iteration as the number of processes varies. It can be seen from the curves that the training time per iteration decreases dramatically at first and turns smooth afterwards as the number of



(a) PNMF

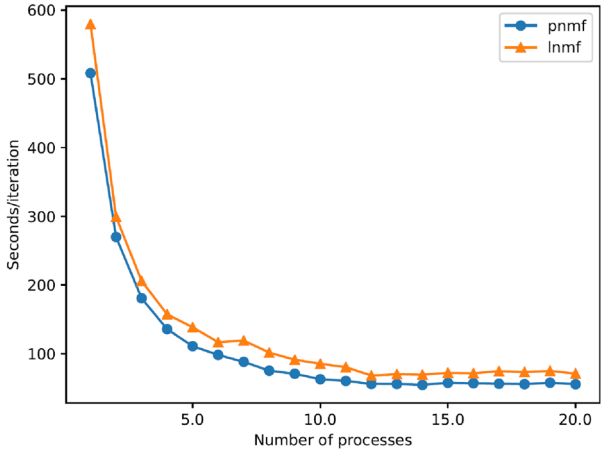


(b) LNMF

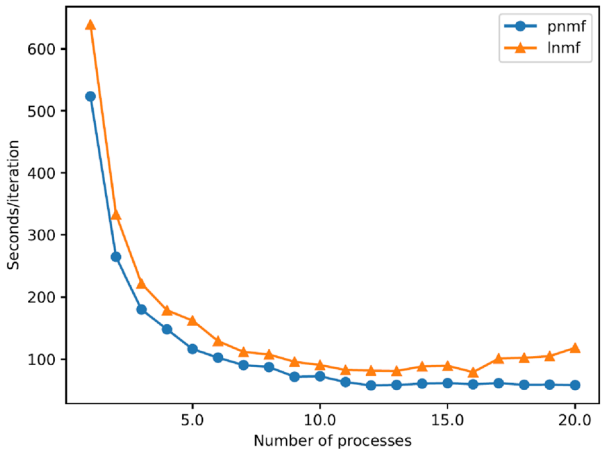
Fig. 1 Convergence time of learning algorithm on SNAP datasets

processes increases. The significant drop when the number of processes is less than 10 validates the correctness of our parallel paradigm. When the number of processes is greater than 10, the training time hardly decreases or even increases a bit because the synchronization after each iteration becomes the bottleneck.

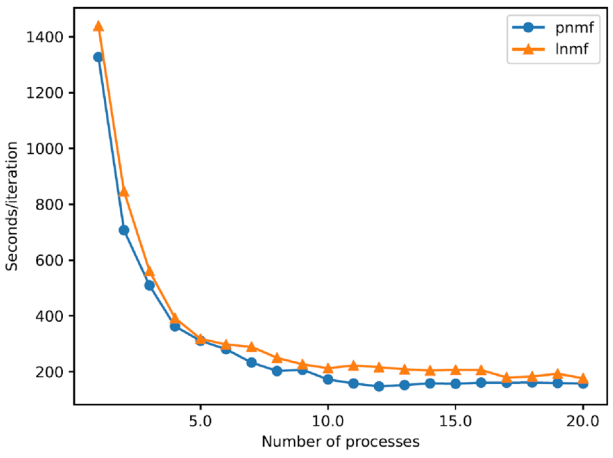
As for the quality of detected communities, we do not show the comparison but the parallel version achieves nearly the same F_1 score as the single process one regardless of the number of processes we use. The loss drops at a similar pace as well. To summarize, the parallel paradigm of our learning algorithm not only decreases the running time, but also preserves the the quality of detected communities.



(a) Amazon



(b) DBLP



(c) YouTube

Fig. 2 Comparison of single-process implementation with parallel implementation in terms of time on SNAP datasets

7 Conclusion and future work

In this paper, we incorporate both link preference information and locality information into overlapping community detection. First, we propose a *Preference-based Non-negative Matrix Factorization* (*PNMF*) model with an intuition that a node prefers its neighbors than its non-neighbors. We maximize the likelihood of a preference order for each node instead of simply approximating the original adjacency matrix in value. Our PNMF model can eliminate the unreasonable indiscriminate penalty on pairs inside and between communities. Second, we propose a *Locality-based Non-negative Matrix Factorization* (*LNMF*) model to further improve the performance of PNMF model with the help of locality. We exploit local area around a node formally defined as *K*-degree local network to enhance the previous preference system. In details, we extend a two-level preference system which only distinguish neighbors and non-neighbors to a three-level preference system which split the set of non-neighbors into local non-neighbors and distant non-neighbors. In parameter learning of both models, we employ stochastic gradient descent with bootstrap sampling to learn the parameters of node-community membership matrix. Experiments on several benchmark datasets including large ones with ground-truth communities show that both models outperform state-of-art approaches under multiple metrics and the LNMF model achieves better results than the PNMF model.

Currently, our work only employs binary networks as input, but it can be extended to multiplex networks as well. In multiplex networks, all layers share the same set of nodes but may have different topology, and inter-layer edges are allowed (Boccaletti et al. 2014). For each layer, we can apply either PNMF or LNMF model to obtain an objective function. Since all layers share the same node set, we can simply sum up all the objectives functions to learn via stochastic gradient descent. However, such extension cannot deal with inter-layer edges.

Acknowledgement The work described in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CUHK 14208815 and No. CUHK 14205214 of the General Research Fund), and 2018 Microsoft Research Asia Collaborative Research Award.

References

Ahn YY, Bagrow JP, Lehmann S (2010) Link communities reveal multiscale complexity in networks. Nature 466(7307):761–764

Airoldi EM, Blei DM, Fienberg SE, Xing EP (2008) Mixed membership stochastic blockmodels. J Mach Learn Res 9(1981–2014):3

Boccaletti S, Bianconi G, Criado R, Del Genio CI, Gómez-Gardenes J, Romance M, Sendina-Nadal I, Wang Z, Zanin M (2014) The structure and dynamics of multilayer networks. Phys Rep 544(1):1–122

Chen W, Liu Z, Sun X, Wang Y (2010) A game-theoretic framework to identify overlapping communities in social networks. Data Min Knowl Discov 21(2):224–240

Clauset A, Newman ME, Moore C (2004) Finding community structure in very large networks. Phys Rev E 70(6):066111

Coscia M, Rossetti G, Giannotti F, Pedreschi D (2012) Demon: a local-first discovery method for overlapping communities. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 615–623

Danon L, Diaz-Guilera A, Duch J, Arenas A (2005) Comparing community structure identification. J Stat Mech Theory Exp 09:P09008

Duch J, Arenas A (2005) Community detection in complex networks using extremal optimization. Phys Rev E 72(2):027104

Fortunato S (2010) Community detection in graphs. Phys Rep 486(3):75–174

Fortunato S, Hric D (2016) Community detection in networks: a user guide. Phys Rep 659:1–44

Girvan M, Newman ME (2002) Community structure in social and biological networks. Proc Natl Acad Sci 99(12):7821–7826

Guimera R, Sales-Pardo M, Amaral LAN (2004) Modularity from fluctuations in random graphs and complex networks. Phys Rev E 70(2):025101

Jin D, Chen Z, He D, Zhang W (2015) Modeling with node degree preservation can accurately find communities. In: 29th AAAI conference on artificial intelligence. AAAI Press, pp 160–167

Kumpula JM, Kivelä M, Kaski K, Saramäki J (2008) Sequential algorithm for fast clique percolation. Phys Rev E 78(2):026109

Lancichinetti A, Fortunato S (2009) Community detection algorithms: a comparative analysis. Phys Rev E 80(5):056117

Lancichinetti A, Fortunato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. Phys Rev E 78(4):046110

Lancichinetti A, Fortunato S, Kertész J (2009) Detecting the overlapping and hierarchical community structure in complex networks. New J Phys 11(3):033015

Leskovec J, Lang KJ, Mahoney M (2010) Empirical comparison of algorithms for network community detection. In: Proceedings of the 19th international conference on world wide web. ACM, New York, pp 631–640

Li Y, He K, Kloster K, Bindel D, Hopcroft J (2018) Local spectral clustering for overlapping community detection. ACM Trans Knowl Discov Data (TKDD) 12(2):17

Lin CJ (2007) Projected gradient methods for nonnegative matrix factorization. Neural Comput 19(10):2756–2779

McAuley J, Leskovec J (2012) Learning to discover social circles in ego networks. In: Advances in neural information processing systems 25 (NIPS 2012), pp 548–556

McDaid AF, Greene D, Hurley N (2011) Normalized mutual information to evaluate overlapping community finding algorithms. arXiv preprint arXiv:http://arxiv.org/abs/1110.2515

McPherson M, Smith-Lovin L, Cook JM (2001) Birds of a feather: homophily in social networks. Annu Rev Sociol 27(1):415–444

Newman ME (2001) The structure of scientific collaboration networks. Proc Natl Acad Sci 98(2):404–409

Newman ME (2004) Fast algorithm for detecting community structure in networks. Phys Rev E 69(6):066133

Newman ME (2006) Modularity and community structure in networks. Proc Natl Acad Sci 103(23):8577–8582

Palla G, Derényi I, Farkas I, Vicsek T (2005) Uncovering the overlapping community structure of complex networks in nature and society. Nature 435(7043):814–818

Psorakis I, Roberts S, Ebden M, Sheldon B (2011) Overlapping community detection using Bayesian non-negative matrix factorization. Phys Rev E 83(6):066114

Recht B, Re C, Wright S, Niu F (2011) Hogwild: a lock-free approach to parallelizing stochastic gradient descent. In: Advances in neural information processing systems 24 (NIPS 2011), pp 693–701

Rendle S, Balby Marinho L, Nanopoulos A, Schmidt-Thieme L (2009a) Learning optimal ranking with tensor factorization for tag recommendation. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 727–736

Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2009b) BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the 25th conference on uncertainty in artificial intelligence. AUAI Press, pp 452–461

Wang F, Li T, Wang X, Zhu S, Ding C (2011) Community discovery using nonnegative matrix factorization. Data Min Knowl Discov 22(3):493–521

Whang JJ, Gleich DF, Dhillon IS (2016) Overlapping community detection using neighborhood-inflated seed expansion. IEEE Trans Knowl Data Eng 28(5):1272–1284

Wu W, Kwong S, Zhou Y, Jia Y, Gao W (2018) Nonnegative matrix factorization with mixed hypergraph regularization for community detection. Inf Sci 435:263–281

Xie J, Kelley S, Szymanski BK (2013) Overlapping community detection in networks: the state-of-the-art and comparative study. ACM Comput Surv (CSUR) 45(4):43

Yang J, Leskovec J (2015) Defining and evaluating network communities based on ground-truth. Knowledge and Information Systems 42(1):181–213

Yang J, Leskovec J (2013) Overlapping community detection at scale: a nonnegative matrix factorization approach. In: Proceedings of the sixth ACM international conference on web search and data mining. ACM, New York, pp 587–596

Zhang Y, Yeung DY (2012) Overlapping community detection via bounded nonnegative matrix tri-factorization. In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, New York, pp 606–614

Zhang H, King I, Lyu MR (2015a) Incorporating implicit link preference into overlapping community detection. In: Proceedings of the 29th AAAI conference on artificial intelligence. AAAI Press, pp 396–402

Zhang H, Lyu MR, King I (2015b) Exploiting k-degree locality to improve overlapping community detection. In: Proceedings of the 24th international joint conference on artificial intelligence (IJCAI 2015), pp 2394–2400

Zhao T, McAuley J, King I (2014) Leveraging social connections to improve personalized ranking for collaborative filtering. In: Proceedings of the 23rd ACM international conference on conference on information and knowledge management. ACM, New York, pp 261–270