Symmetric Nonnegative Matrix Factorization-Based Community Detection Models and Their Convergence Analysis

Project Review

A.Saicharan—D. Obileshsai—R Reddappa— Mrudula A

Roll No.: EC21B1035 — EC21B1033 — EC21B1114— EC21B1115

Team no: 19

Indian Institute of Information Technology
Design and Manufacturing, Chennai, Tamilnadu-600127, India

Presentation Outline

Introduction

- Networks, such as social networks or biological networks, are often represented as matrices where rows and columns correspond to nodes, and the matrix entries capture the relationships or interactions between nodes.
- Networks can be large and sparse, especially in the case of social networks or biological systems.
- Matrix factorization allows for the reduction of the dimensionality of the network while preserving essential structural information. This is achieved by decomposing the original network matrix into two lower-rank matrices.

Introduction

Community detection is a fundamental problem in network analysis that aims to partition nodes in a network into groups or communities based on their connectivity patterns. Symmetric Nonnegative Matrix Factorization (SNMF) is a popular approach for community detection due to its ability to handle non-negative data.

In this paper, the authors propose innovative scaling-factor-adjusted NMU schemes for SNMF-based community detection models to achieve highly accurate community detection

Problem definition

- Given a symmetric matrix $A_{n\times n}$, an SNMF model seeks for its low-rank approximation $\hat{A}$ on the latent factor (LF) matrix $X_{n\times K}$ with K denoting the dimension of the latent feature space and $K \leq n$, i.e., $\hat{A} = XX^T$. To obtain X, an objective function describing the difference between A and $\hat{A}$ is necessary, as the following Euclidean distance function:

$$\min_X \text{JSNMF} = \min_X \|A - XX^T\|_F^2, \quad \text{s.t. } X \geq 0 \tag{1}$$

where the operator $\|\cdot\|_F$ computes the Frobenius norm of an enclosed matrix.

Objective

- The main objective is to minimize the objective function min

$$\min_X J_{\text{SNMF}} = \min_X \left\| A - XX^T \right\|_{\mathrm{F}}^2, \quad \text{s.t. } X \geq 0$$

by proposing a scaling-factor-adjusted NMU schemes for SNMF-based community detection models to achieve highly accurate community detection with theoretically guaranteed convergence behaviors. Than the existing standard NMU

$$X_{ik} \leftarrow X_{ik} \frac{(AX)_{ik}}{(XX^T X)_{ik}}$$

that frequently makes an SNMF model suffer from training fluctuations caused by the scaling factor $(AX)_{ik}/(XX\ \mathrm{T}X)_{ik}$. It can become too aggressive to enable a steady training process.

Methods to achieve the objective I

- This study proposes scaling-factor-adjusted NMU (SNMU) schemes for SNMF and graph-regularized SNMF (GSNMF) models, thereby achieving highly accurate community detectors with theoretically guaranteed convergence behaviors.
- Four novel community detection models have been presented, i.e., α-SNMF, β-SNMF, α-GSNMF, and β-GSNMF.

Methods to achieve the objective II

Let $M^{n \times K}$ be a matrix caching the scaling factors for all parameters in X, this study proposes the following tuning rules for learning rate equation of SNMF:

Non-linear tuning: $M_{ik} = \left(\frac{(AX)_{ik}}{(XX^T X)_{ik}} \right)^{\alpha}, \quad 0 < \alpha \leq 1$

Linear tuning: $M_{ik} = \beta \frac{(AX)_{ik}}{(XX^T X)_{ik}}, \quad 0 < \beta \leq 1.$

Traditional non-negative matrix factorization methods like Alpha SNMF and Beta SNMF may struggle with detecting overlapping communities where nodes belong to multiple communities simultaneously. GSNMF can incorporate regularization terms that promote a more flexible assignment of nodes to multiple

Methods to achieve the objective III

communities, allowing for better representation of overlapping structures.,

$\min_X J_{\mathrm{GSNMF}} = \min_X \left\|A - XX^T\right\|_{\mathrm{F}}^2 + \lambda \operatorname{tr}\left(X^T L X\right)$

s.t. $X \geq 0$

let λ is the graph regularization constant.

Non-linear tuning: $M_{ik} = \left(\frac{(1+\lambda)(AX)_{ik}}{\left(XX^T X+\lambda DX\right)_{ik}}\right)^{\alpha}, \quad 0 < \alpha \leq 1$

Linear tuning: $M_{ik} = \beta \frac{(1+\lambda)(AX)_{ik}}{\left(XX^T X+\lambda DX\right)_{ik}}, \quad 0 < \beta \leq 1.$

Action Plan

1. Understand the problem of community detection and the limitations of existing methods.
2. Study the proposed scaling-factor-adjusted NMU schemes and choose the appropriate one for SNMF-based community detection models.
3. Apply the chosen scheme to SNMF or GSNMF models to achieve a novel SNMF-based community detector.
4. Set the hyperparameters of the model and validate its performance using a set of experiments.
5. Conduct theoretical studies to show that the model can keep its loss function non-increasing during its training process and converge to a stationary point.
6. Conduct empirical studies on real-world social networks to show the effectiveness of the proposed methods

Simulation Setup And Outputs

1. Python FrameWorks Like Numpy and Matplotlib are used.

```python
def SNMFalpha(A,k,max_iter,alpha):
    X = random_initialization(A,k)
    e = 1.0e-10
    err = []
    for n in range(max_iter):
        # Update X
        AX = A@X
        XX_TX = X@X.T@X
        for i in range(np.size(X, 0)):
            for j in range(np.size(X, 1)):
                X[i, j] = X[i, j] * (((AX[i, j]) / (XX_TX[i,j]))**alpha)
        err.append(np.sum(np.square(A - X@X.T))/9)
        if(n%10 == 0):
            print(f"L2 Norm for {n} iteration : {err[n]} ")
    return X,err
```

```python
def SNMFbeta(A,k,max_iter,beta):
    X = random_initialization(A,k)
    e = 1.0e-10
    err = []
    for n in range(max_iter):
        # Update X
        AX = A@X
        XX_TX = X@X.T@X
        for i in range(np.size(X, 0)):
            for j in range(np.size(X, 1)):
                X[i, j] = X[i, j] * ((1 - beta) + (beta*(AX[i, j]) / (XX_TX[i,j])))
        err.append(np.sum(np.square(A - X@X.T))/9)
        if(n%10 == 0):
            print(f"L2 Norm for {n} iteration : {err[n]} ")
    return X,err
```

```python
def GSNMFalpha(A,k,max_iter,lambd,alpha):
    X = random_initialization(A,k)
    e = 1.0e-10
    err = []
    D = Diagonal_Matrix(A)
    for n in range(max_iter):
        # Update X
        AX = A@X
        XX_TX = X@X.T@X
        DX = D@X
        for i in range(np.size(X, 0)):
            for j in range(np.size(X, 1)):
                X[i, j] = X[i, j] * ((((1 + lambd)*AX[i, j]) / (XX_TX[i,j]+ lambd*DX[i,j]))**alpha)
        err.append(np.sum(np.square(A - X@X.T))/9)
        if(n%10 == 0):
            print(f"L2 Norm for {n} iteration : {err[n]} ")
    return X,err
```

```python
def GSNMFbeta(A,k,max_iter,lambd,beta):
    X = random_initialization(A,k)
    e = 1.0e-10
    err = []
    D = Diagonal_Matrix(A)
    for n in range(max_iter):
        # Update X
        AX = A@X
        XX_TX = X@X.T@X
        DX = D@X
        for i in range(np.size(X, 0)):
            for j in range(np.size(X, 1)):
                X[i, j] = X[i, j] * ((1 - beta) + (beta*((1 + lambd)*AX[i, j]) / (XX_TX[i,j]+ lambd*DX[i,j])))
        err.append(np.sum(np.square(A - X@X.T))/9)
        if(n%10 == 0):
            print(f"L2 Norm for {n} iteration : {err[n]} ")
    return X,err
```

Figure: Simulation Setup

Simulation Setup And Outputs

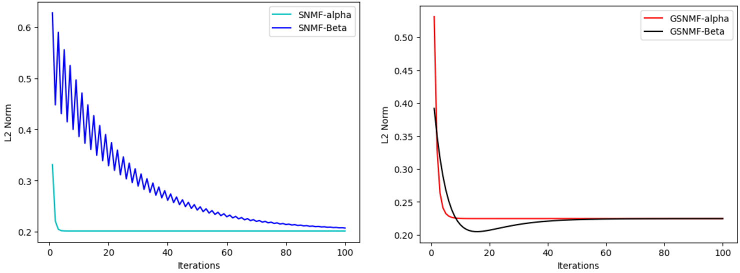

Figure: Outputs

Conclusions

1. On most cases, alpha-SNMF beats beta-SNMF and alpha-GSNMF beats beta-GSNMF when addressing the task of community detection. In particular, alpha-GSNMF significantly outperforms all of its peers in terms of detection accuracy
2. Optimal alpha and beta can be detected through a parameter sensitivity test on probe sets, making the proposed four models practical for industrial applications.
3. Considering alpha-SNMF and alpha-GSNMF models, the optimal alpha changes sharply on different data sets. Considering beta-SNMF and beta-GSNMF models, their highest detection accuracy occurs as beta = 0.99. The reason for them remains unveiled.