

Article

Hierarchical Object Part Learning Using Deep L_p Smooth Symmetric Non-Negative Matrix Factorization

Shunli Li ^{1,2,†} , Chunli Song ^{1,†}, Linzhang Lu ^{1,3,*,†} and Zhen Chen ^{1,*,†}

¹ School of Mathematical Sciences, Guizhou Normal University, Guiyang 550025, China; 20030060022@gznu.edu.cn (S.L.); chunlison@git.edu.cn (C.S.)

² College of Mathematics and Information Science, Guiyang University, Guiyang 550005, China

³ School of Mathematical Sciences, Xiamen University, Xiamen 361005, China

* Correspondence: llz@xmu.edu.cn or llz@gznu.edu.cn (L.L.); zchen@gznu.edu.cn (Z.C.)

† These authors contributed equally to this work.

Abstract: Nowadays, deep representations have gained significant attention due to their outstanding performance in a wide range of tasks. However, the interpretability of deep representations in specific applications poses a significant challenge. For instances where the generated quantity matrices exhibit symmetry, this paper introduces a variant of deep matrix factorization (deep MF) called deep L_p smooth symmetric non-negative matrix factorization (DSSNMF), which aims to improve the extraction of clustering structures inherent in complex hierarchical and graphical representations in high-dimensional datasets by improving the sparsity of the factor matrices. We successfully applied DSSNMF to synthetic datasets as well as datasets related to post-traumatic stress disorder (PTSD) to extract several hierarchical communities. Specifically, we identified non-disjoint communities within the partial correlation networks of PTSD psychiatric symptoms, resulting in highly meaningful clinical interpretations. Numerical experiments demonstrate the promising applications of DSSNMF in fields like network analysis and medicine.

Keywords: deep MF; L_p smooth; symmetric matrix factorization; hierarchical communities



Citation: Li, S.; Song, C.; Lu, L.; Chen, Z. Hierarchical Object Part Learning Using Deep L_p Smooth Symmetric Non-Negative Matrix Factorization. *Symmetry* **2024**, *16*, 312. <https://doi.org/10.3390/sym16030312>

Academic Editor: Alexei Kanel-Belov

Received: 4 February 2024

Revised: 27 February 2024

Accepted: 29 February 2024

Published: 6 March 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Non-negative matrix factorization (NMF) [1] is a valuable technique that has been widely used in various areas of feature learning, including document clustering [2,3], recognition of objects and faces [4,5], signal analysis [6,7], and community detection [8,9], among others.

In the era of artificial intelligence, extracting meaningful features from massive and complex data is a significant challenge. Consequently, matrix factorization has gained significant prominence as a technique for extracting features and reducing dimensionality in recent years. NMF is a technique used to approximately decompose a non-negative data matrix $\mathbf{X} \in \mathbb{R}_+^{n \times m}$ into the product of two non-negative factorization matrices $\mathbf{U} \in \mathbb{R}_+^{n \times r}$ and $\mathbf{V} \in \mathbb{R}_+^{r \times m}$, with the objective of approximating $\mathbf{X} \approx \mathbf{UV}$. Here, $\mathbf{X}$ represents a data matrix where each column corresponds to a data point in dimension m , and r denotes the factorization rank, indicating that each data point can be expressed as a linear combination of r basis vectors. In order to improve feature extraction and data compression, researchers have developed several variants of NMF. These variants introduce different constraints on the basis matrix $\mathbf{U}$ and the coefficient matrices $\mathbf{V}$. Some examples of these constraints include non-negativity [8], sparsity [10], manifold constraints [4], non-smoothness [5], and orthogonality [11]. When the matrix $\mathbf{X}$ represents the similarity values between two data points and exhibits symmetry (i.e., $\mathbf{X} = \mathbf{X}^T \in \mathbb{R}^{n \times n}$ and $\mathbf{V} = \mathbf{U}^T$), symmetric non-negative matrix factorization [12] is proposed as a suitable approach. This method is commonly applied to matrices such as word co-occurrence matrices in topic modeling [13,14] or adjacency matrices of undirected graphs. It is worth noting

that while the input symmetric matrix $\mathbf{X}$ typically has non-negative values, it is not strictly required. Nonetheless, it is essential to highlight that the resulting factor matrices must be non-negative.

Although NMF delivers excellent results in feature extraction and data compression, researchers continue to explore the presence of hierarchical information within objects. More recently, advancements in matrix factorization techniques, such as multilayer MF [15–18] and deep MF [18], influenced by deep learning theory, have elevated the field to a new level. Deep non-negative matrix factorization (deep NMF) builds upon standard NMF and approximates it as a product of multiple factor matrices. More precisely, the matrix $\mathbf{X}$ undergoes L layers of factorizations resulting in an approximation represented by $\mathbf{X}$ such that $\mathbf{X} \approx \mathbf{U}_L \mathbf{V}_L \mathbf{V}_{L-1} \cdots \mathbf{V}_1$. The hierarchical form of equivalence can be expressed as follows:

$$\begin{aligned}\mathbf{X} &\approx \mathbf{U}_1 \mathbf{V}_1, \\ \mathbf{U}_1 &\approx \mathbf{U}_2 \mathbf{V}_2, \\ &\dots \\ \mathbf{U}_{L-1} &\approx \mathbf{U}_L \mathbf{V}_L.\end{aligned}\tag{1}$$

where $\mathbf{U}_l \in \mathbb{R}_+^{m \times r_l}$ and $\mathbf{V}_l \in \mathbb{R}_+^{r_l \times r_{l-1}}$ ($l = 1, \dots, L$) with $r_0 = n$. In scheme of (1) the dimension r_l of each layer is the rank of the factorization at layer l , and r_l ($l = 1, \dots, L$) are assumed to be decreasing, that is, $r_1 > r_2 > \dots > r_L$, see [19,20] for more details.

Deep MF has the capability to conduct the hierarchical decomposition of intricate input datasets, enhancing the interpretability of the dataset. It is widely employed in various domains, including hierarchical feature extraction [21], recommender systems [17], and multi-view clustering [22,23].

In the existing literature, the combination of smoothness and symmetry in deep NMF has not been explored. We will investigate a deep NMF, namely deep L_p smooth symmetric non-negative matrix factorization (DSSNMF). The main contributions and motivations for the algorithm are as follows:

- We investigate the case where the input quantity matrix $\mathbf{X}$ is a symmetric matrix, such as the adjacency matrix of an undirected graph. If $\mathbf{X}$ is constructed correctly, the approximate deep factorization of $\mathbf{X}$ generates a series of non-negative clustering assignment matrices that effectively capture the inherent clustering structure of each layer. Numerical experiments conducted on synthetic datasets with complex structures demonstrate that the algorithm provides good interpretability of the hierarchical clustering information.
- In order to obtain smoother and more accurate solutions, we introduce the L_p constraints of matrices into the framework of depth-symmetric non-negative matrix factorization and propose the DSSNMF method. The method fully preserves the smooth and more accurate optimized solution.
- In this paper, we investigate the combination of smooth and symmetry in deep matrix factorization through the proposed DSSNMF algorithm. The motivation behind this research is to explore the hierarchical structure present in complex datasets by enhancing the sparsity of the factor matrices. Our results demonstrate the effectiveness of DSSNMF in extracting hierarchical communities both in synthetic datasets and in post-traumatic stress disorder (PTSD) datasets, specifically revealing non-disjoint communities in the PTSD psychiatric symptom partial correlation networks. Moreover, our numerical experiments highlight the promising applications of DSSNMF in fields such as network analysis and medicine.

This paper is organized into five sections. Section 2 shows the process of the construction of the DSSNMF model. The DSSNMF algorithm and its implementation conditions are presented in Section 3. To demonstrate the performance of the DSSNMF algorithm,

experiments are conducted on both the PTSD dataset and the simulated dataset in Section 4. Finally, Section 5 provides a conclusion.

2. DSSNMF Modeling

In this study, matrices are represented by bold capital letters indicated earlier. For example, $\mathbf{U}$ or $\mathbf{U}_l$ represents a matrix, $\mathbf{U}(i, j)$ indicates the element at position (i, j) in the matrix, $\mathbf{U}(:, j)$ and $\mathbf{U}(i, :)$ represent the j -th column and i -th row of the matrix, respectively. The symbols $(\cdot)^\top$ and $\text{Tr}(\cdot)$ denote the transpose and trace of the matrix, and $\langle \cdot, \cdot \rangle$ and $\bar{\cdot}$ indicate the scalar product of two vectors and the mean of a vector, respectively.

To make the modeling process understandable, we take the example of a simple undirected graph. Let $\mathbf{X} \in \mathbb{R}^{n \times n}$ be a symmetric adjacency matrix of a graph (see Figure 1). The element at position (i, j) in $\mathbf{X}$ represents the similarity value between data points $\mathbf{x}_i$ and $\mathbf{x}_j$, which correspond to nodes. The objective of DSSNMF is to use L layers of factorizations to obtain, at each layer l , a non-negative symmetric approximation of rank r_l for the original matrix $\mathbf{X}$. Specifically, at the first layer, we have $\mathbf{X} \approx \mathbf{U}_1 \mathbf{U}_1^\top$, where $\mathbf{U}_1 \in \mathbb{R}_+^{n \times r_1}$, similar to symNMF [12]. At the second layer, we factorize the matrix $\mathbf{U}_1$ as $\mathbf{U}_1 \approx \mathbf{U}_2 \mathbf{V}_2$ with $\mathbf{U}_2 \in \mathbb{R}_+^{n \times r_2}$ and $\mathbf{V}_2 \in \mathbb{R}_+^{r_2 \times r_1}$, where $r_2 < r_1$. This results in a new symmetric approximation of rank $r_2 < r_1$ for $\mathbf{X}$, namely $\mathbf{X} \approx \mathbf{U}_2 \mathbf{V}_2 \mathbf{V}_2^\top \mathbf{U}_2^\top$. Similarly, the L -th layer is decomposed as: $\mathbf{X} \approx \mathbf{U}_L \mathbf{V}_L \cdots \mathbf{V}_2 \mathbf{V}_2^\top \cdots \mathbf{V}_L^\top \mathbf{U}_L^\top$, with $\mathbf{U}_l \in \mathbb{R}_+^{n \times r_l}$ and $\mathbf{V}_l \in \mathbb{R}_+^{r_l \times r_{l-1}}$, ($l = 1, \dots, L$), $r_L < r_{L-1} < \dots < r_2 < r_1$. This is how the deep symmetric matrix factorization model is constructed. From the perspective of the entire decomposition process, in simple terms, the decomposition factor $\mathbf{U}_1$ can be understood as comprising r_1 communities, each containing n points, and $\mathbf{U}_1(i, j)$ refers to the i -th point belonging to the j -th community. Furthermore, as the factorization continues and the rank decreases, the columns of $\mathbf{U}_2$ can be interpreted as representing the membership degrees of the n data points in r_2 communities. The smaller communities gradually merge into larger ones as the factorization progresses.

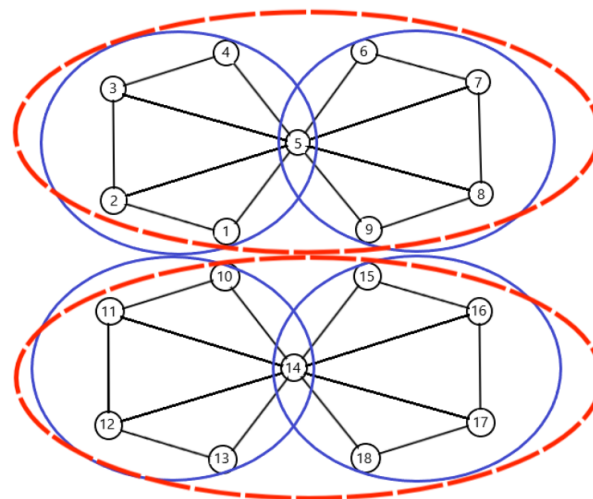


Figure 1. Simple undirected graphs with two layers of communities when $r_1 = 4$ and $r_2 = 2$. The first layer of communities corresponds to the set of nodes surrounded by thin blue solid line ellipses. The second layer of communities corresponds to the set of nodes surrounded by the red thick dashed line ellipse.

To generate smoother and more accurate solutions at each layer of the approximate factorization, we employ a series of additional L_p smoothing constraints with the goal of enhancing the accuracy of the subsequent factorizations. The interpretation of matrix factorization at each layer has two aspects: utilizing linear combinations of basis vectors enables an accurate reconstruction of the data, and the interpretation of the basis vectors is clear-cut.

The rationale behind this approach is to find a method to decompose the data into mutually exclusive components, thereby enhancing the accuracy of the subsequent factorizations.

Nevertheless, deep NMF lacks consideration for the smoothing of the basis matrices in each layer of the factorization, which restricts its ability to generate solutions that are both smooth and accurate. To address this limitation, incorporating L_p smoothing constraints into the deep symmetric NMF model offers a promising approach for achieving a smoother and more accurate solution to the optimization problem. The new deep NMF model is as follows:

The first layer of factorization is: $\mathbf{X} \approx \mathbf{U}_1 \mathbf{U}_1^\top + 2\beta \|\mathbf{U}_1\|_p^p$, where $\mathbf{U}_1 \in \mathbb{R}_+^{n \times r_1}$,

The second layer of factorization is: $\mathbf{X} \approx \mathbf{U}_2 \mathbf{V}_2 \mathbf{V}_2^\top \mathbf{U}_2^\top + 2\beta \|\mathbf{U}_2\|_p^p$, where $\mathbf{U}_2 \in \mathbb{R}_+^{n \times r_2}$, $\mathbf{V}_2 \in \mathbb{R}_+^{r_2 \times r_1}$,

Similarly, the remaining layers can be given in the same way, the L -th layer of factorization is: $\mathbf{X} \approx \mathbf{U}_L \mathbf{V}_L \cdots \mathbf{V}_2 \mathbf{V}_2^\top \cdots \mathbf{V}_L^\top \mathbf{U}_L^\top + 2\beta \|\mathbf{U}_L\|_p^p$, where $\mathbf{U}_L \in \mathbb{R}_+^{n \times r_L}$, $\mathbf{V}_L \in \mathbb{R}_+^{r_L \times r_{L-1}}$.

Where β is the regularization parameter, $\|\mathbf{U}_l\|_p = (\sum_{i,j} |\mathbf{U}_l(i,j)|^p)^{\frac{1}{p}}$, $l = 1, \dots, L$, p is smooth parameter which ranges: $1 < p < 2$. This is the model for *deep L_p smooth symmetric non-negative matrix factorization* (DSSNMF) proposed in this paper.

3. Algorithm for DSSNMF

Given a non-negative symmetric matrix $\mathbf{X} \in \mathbb{R}^{n \times n}$, the standard symNMF involves solving the following optimization problem:

$$\min_{\mathbf{U} \in \mathbb{R}_+^{n \times r}} \|\mathbf{X} - \mathbf{U} \mathbf{U}^\top\|_F^2. \quad (2)$$

In order to design an effective alternating type algorithm for standard symNMF, Li, X. et al. [24] transfer this problem to a penalized asymmetric NMF in the following form:

$$\min_{\mathbf{U} \in \mathbb{R}_+^{n \times r}, \mathbf{V} \in \mathbb{R}_+^{r \times n}} \|\mathbf{X} - \mathbf{U} \mathbf{V}\|_F^2 + \gamma \|\mathbf{U} - \mathbf{V}^\top\|_F^2. \quad (3)$$

where the penalty term $\gamma \|\mathbf{U} - \mathbf{V}\|_F^2$ is included to encourage similarity between the two factors $\mathbf{U}$ and $\mathbf{V}^\top$. For sufficiently large γ , it has been shown that $\mathbf{U} = \mathbf{V}^\top$ holds for the critical points of (3) [24].

Inspired by the work of [20,25], we combine the deep L_p smooth symmetric matrix factorization model, as described in Section 2, with (3) to formulate a comprehensive loss function that incorporates weighted sums of the layer-wise contributions.

$$\begin{aligned} \ell_{DSSNMF} = & \frac{1}{2} (\|\mathbf{X} - \mathbf{U}_1 \mathbf{V}_1\|_F^2 + \gamma_1 \|\mathbf{U}_1 - \mathbf{V}_1^\top\|_F^2 + 2\beta \|\mathbf{U}_1\|_p^p) \\ & + \lambda_1 (\|\mathbf{U}_1 - \mathbf{U}_2 \mathbf{V}_2\|_F^2 + \gamma_2 \|\mathbf{U}_2 - (\mathbf{V}_2 \mathbf{V}_1)^\top\|_F^2 + \beta \|\mathbf{U}_2\|_p^p) \\ & + \cdots + \lambda_{L-1} (\|\mathbf{U}_{L-1} - \mathbf{U}_L \mathbf{V}_L\|_F^2 \\ & + \gamma_L \|\mathbf{U}_L - (\mathbf{V}_L \mathbf{V}_{L-1} \cdots \mathbf{V}_2 \mathbf{V}_1)^\top\|_F^2 + 2\beta \|\mathbf{U}_L\|_p^p). \end{aligned} \quad (4)$$

This loss function represents the error in the execution of the symNMF at each layer, where the weighting parameter, denoted as λ_l , determines the relative importance of each layer. Remarkably, this approach can be viewed as an extension of the multilayer symmetric NMF model proposed by Cichocki et al. [15]. In this extension, instead of considering the sum of errors from individual layer factorizations, we calculate the overall weighted error that accounts for the errors across all layer factorizations. The error for each layer is $Er(l) = \lambda_{l-1} (\|\mathbf{U}_{l-1} - \mathbf{U}_l \mathbf{V}_l\|_F^2 + \gamma_l \|\mathbf{U}_l - (\mathbf{V}_l \mathbf{V}_{l-1} \cdots \mathbf{V}_2 \mathbf{V}_1)^\top\|_F^2 + 2\beta \|\mathbf{U}_l\|_p^p)$ for $l = 1, \dots, L$, with $\mathbf{U}_0 = \mathbf{X}$ and $\lambda_0 = 1$. The error $Er(l)$ at each layer consists of three components. The first component represents the reconstruction error between $\mathbf{U}_{l-1}$ and $\mathbf{U}_l \mathbf{V}_l$, capturing the quality of the rank r_l approximation. The second component

ensures that the factorization at each layer adheres to the symNMF framework. Lastly, the third component is the L_p constraint term. Therefore, both (4) and (3) adopt a similar method to solve non-quadratic optimization subproblems. This allows us to derive iterative updating rules for the factor matrices $\mathbf{U}_l$ and $\mathbf{V}_l$ at each layer. These matrices are alternately optimized, resulting in a monotonically decreasing objective function.

To minimize (4), the block coordinate descent (BCD) [15,20,25] is employed, necessitating the blockwise updating of the factor matrices. In this study, we utilize the fast projected gradient method (FPGM) [26] to update the factor matrices $\mathbf{U}_l$ or $\mathbf{V}_l$. FPGM is a widely recognized first-order optimization technique that can efficiently update any $\mathbf{U}_l$ or $\mathbf{V}_l$, $l = 1, \dots, L$, as outlined in Algorithm 1. Considering the convexity of each layer's subproblems, we select $\frac{1}{L}$ as the step size, where L represents the Lipschitz constant. Here the symbol $\mathbb{P}_{\mathbb{U}}$ denotes the projection on the feasible set $\mathbb{U}$.

Algorithm 1 Restarted fast projected gradient method (FPGM) for $\mathbf{U}$

Require: $\mathbf{U}^{(0)}$: Initial matrix, $\mathbb{U}$: Feasible set, $\ell(\mathbf{U})$: Loss function, $\alpha_1 \in (0, 1)$: Parameter

Ensure: $\mathbf{U}$ that minimizes $\ell(\mathbf{U})$ such that $\ell(\mathbf{U}) < \ell(\mathbf{U}^{(0)})$

- 1: Compute Lipschitz constant L of ℓ , $M = \mathbf{U}^{(0)}$
 - 2: **for** $k = 1$ to N **do** %Number of iterations
 - 3: $\mathbf{U}^{(k)} = \mathbb{P}_{\mathbb{U}}(M - \frac{1}{L} \nabla \ell(\mathbf{U}))$
 - 4: $M = \mathbf{U}^{(k)} + \varepsilon_k(\mathbf{U}^{(k)} - \mathbf{U}^{(k-1)})$ with $\varepsilon_k = \frac{\alpha_k(1 - \alpha_k)}{\alpha_k^2 + \alpha_{k+1}}$, $\alpha_{k+1} = \frac{1}{2}(\sqrt{\alpha_k^4 + 4\alpha_k^2} - \alpha_k^2)$
 - 5: **if** $\ell(\mathbf{U}^{(k)}) > \ell(\mathbf{U}^{(k-1)})$ **then**
 - 6: $M = \mathbf{U}^{(k-1)}$, $\alpha_{k+1} = \alpha_1$
-

Algorithm 1 requires the gradient of the loss function (4) with respect to $\mathbf{U}_l$ and $\mathbf{V}_l$ ($l = 1, \dots, L$). By introducing $\mathbf{U}_0 = \mathbf{X}$, $\lambda_0 = \gamma_0 = 1$, it is straightforward to derive

$$\begin{aligned}\frac{\partial \ell_{DSSNMF}}{\partial \mathbf{U}_l} &= \frac{\partial Er(l)}{\partial \mathbf{U}_l}, \\ \frac{\partial \ell_{DSSNMF}}{\partial \mathbf{V}_l} &= \frac{\partial Er(l)}{\partial \mathbf{V}_l}.\end{aligned}$$

On the other hand, $Er(l)$ can be deformed as

$$\begin{aligned}Er(l) &= \lambda_{l-1}(\|\mathbf{U}_{l-1} - \mathbf{U}_l \mathbf{V}_l\|_F^2 + \gamma_l \|\mathbf{U}_l - (\mathbf{V}_l \mathbf{V}_{l-1} \cdots \mathbf{V}_2 \mathbf{V}_1)^\top\|_F^2 + 2\beta \|\mathbf{U}_l\|_p^p) \\ &= \lambda_{l-1}(tr((\mathbf{U}_{l-1} - \mathbf{U}_l \mathbf{V}_l)(\mathbf{U}_{l-1} - \mathbf{U}_l \mathbf{V}_l)^\top) \\ &\quad + \gamma_l tr((\mathbf{U}_l - (\mathbf{V}_l \mathbf{Q})^\top)(\mathbf{U}_l^\top - \mathbf{V}_l \mathbf{Q})) + 2\beta \|\mathbf{U}_l\|_p^p) \\ &= \lambda_{l-1}((tr(\mathbf{U}_{l-1} \mathbf{U}_{l-1}^\top) - 2tr(\mathbf{U}_{l-1} \mathbf{V}_l^\top \mathbf{U}_l^\top) + tr(\mathbf{U}_l \mathbf{V}_l \mathbf{V}_l^\top \mathbf{U}_l^\top)) \\ &\quad + \gamma_l (tr(\mathbf{U}_l \mathbf{U}_l^\top) - 2tr(\mathbf{U}_l \mathbf{V}_l \mathbf{Q}) + tr(\mathbf{Q}^\top \mathbf{V}_l^\top \mathbf{V}_l \mathbf{Q})) + 2\beta \|\mathbf{U}_l\|_p^p),\end{aligned}$$

where $\mathbf{Q} = \mathbf{V}_{l-1} \cdots \mathbf{V}_2 \mathbf{V}_1$. If we take the derivatives with respect to $\mathbf{U}_l$ and $\mathbf{V}_l$, we obtain

$$\begin{aligned}\frac{\partial \ell_{DSSNMF}}{\partial \mathbf{U}_l} &= \lambda_{l-1}((-2\mathbf{U}_{l-1} \mathbf{V}_l^\top + \mathbf{U}_l \mathbf{V}_l \mathbf{V}_l^\top + \mathbf{U}_l \mathbf{V}_l \mathbf{V}_l^\top) + \gamma_l (2\mathbf{U}_l - 2\mathbf{Q}^\top \mathbf{V}_l^\top) + 2p\beta \mathbf{U}_l^{p-1}) \\ &= 2\lambda_{l-1}((\mathbf{U}_l \mathbf{V}_l \mathbf{V}_l^\top - \mathbf{U}_{l-1} \mathbf{V}_l^\top) + \gamma_l (\mathbf{U}_l - \mathbf{Q}^\top \mathbf{V}_l^\top) + p\beta \mathbf{U}_l^{p-1}) \\ \frac{\partial \ell_{DSSNMF}}{\partial \mathbf{V}_l} &= 2\lambda_{l-1}((\mathbf{U}_l^\top \mathbf{U}_l \mathbf{V}_l - \mathbf{U}_l^\top \mathbf{U}_{l-1}) + \gamma_l (\mathbf{V}_l \mathbf{Q} \mathbf{Q}^\top - \mathbf{U}_l^\top \mathbf{Q}^\top)).\end{aligned}$$

Next, Algorithm 2 gives the algorithm for minimizing the loss function (4) with general constraints on the factor matrices $\mathbf{U}_l$ and $\mathbf{V}_l$.

Algorithm 2 DSSNMF**Input:** Symmetric matrix $\mathbf{X}$ **Output:** Hierarchical factor matrix $\mathbf{U}_1, \dots, \mathbf{U}_L$ and $\mathbf{V}_1, \dots, \mathbf{V}_L$

- 1: Choose the number of factorization layers L , factorization rank per layer $r_1, \dots, r_L$, and initial matrices $\mathbf{U}_l^{(0)}$ and $\mathbf{V}_l^{(0)}$ for each layer, regularization parameter β , smooth parameter p
- 2: **for** $k = 1, \dots$, **do**
- 3: **for** $l = 1, \dots, L$ **do**
- 4: Execute Algorithm 1 and output $\mathbf{V}_l^{(k)}$
- 5: Execute Algorithm 1 and output $\mathbf{U}_l^{(k)}$

Special attention should be given to the selection of hyperparameters, particularly the number of layers L and their factorization rank r_l , which are crucial elements of the deep NMF model (refer to the first step of Algorithm 2). These hyperparameters can be manually specified by the user if there is existing prior knowledge about the network. In line with [20], we utilize the multilayer matrix factorization (MulMF) method to initialize the hierarchical factor matrices $\mathbf{U}_l^{(0)}$ and $\mathbf{V}_l^{(0)}$. Specifically, the L -layer deep L_p smooth symmetric non-negative matrix factorization initializes the factors layer by layer, supplying the initial factor matrix before entering the DSSNMF procedure. If no a priori information about the network is available, we employ the widely recognized Louvain Method (LM) [27], which efficiently identifies modular partitions in large networks, yielding a comprehensive hierarchical community structure and providing various resolutions for community detection. After t iterations, the LM method generates a graph partition consisting of r_t distinct communities, ensuring that each node belongs to only one community. This effectively extracts a bottom-up hierarchy of communities within the graph, allowing different interpretations in each iteration, similar to the MulMF algorithm. The main difference is observed in MulMF, where nodes can belong to multiple communities with varying proportions. Without prior network information to guide the selection of hyperparameters, we set the number of layers L in DSSNMF to be equal to the number of LM iterations t . The rank r_l is determined by the number of layers r_t successively extracted by the LM within the communities.

4. Experiments and Discussions

Deep MF has garnered significant attention in recent years due to its ability to learn hierarchical representations from complex data [23,28,29]. The incorporation of symmetry [23], non-negativity constraints [8], sparsity-constrained [30] and pairwise relationships [31] has significantly improved the ability to capture hierarchical structures and community patterns in complex networks. Additionally, the application of deep architectures has further enhanced the representation learning capabilities, leading to improved clustering and recommendation performances. These advancements open up exciting possibilities for exploring hierarchical information extraction in various domains and pave the way for future research in the field of deep MF.

In this section, we employ the DSSNMF algorithm to analyze both synthetic datasets and real psychiatric network datasets. We conduct numerical experiments to extract latent hierarchical features, aiming to uncover previously unknown hierarchical structures, despite the absence of any prior knowledge. This task presents considerable challenges.

4.1. Synthetic Dataset

From the given loss function 4, it is evident that setting the parameters λ_l ($l = 1, \dots, L - 1$), γ_l ($l = 1, \dots, L$) and β is necessary. These parameters play a crucial role in balancing the significance of each layer and facilitating the progression to the subsequent step. Specifically, λ_l is chosen such that $Er^{(0)}(l)$ is equal for all $l > 0$, that is

$\lambda_l = \frac{Er^{(0)}(l)}{Er^{(0)}(l+1)}$. Similarly, the γ_l values are determined so that $Er_1^{(0)}(l) = Er_2^{(0)}(l)$ for all l , which implies $\gamma_l = \frac{Er_1^{(0)}(l)}{Er_2^{(0)}(l)}$.

For our subsequent experiments, we will employ the following comparison algorithm:

- DSNMF: Deep Symmetric Matrix Factorization, which is discussed in detail in [25].
- Multilayer NMF (MulNMF): This method involves multilayer non-negative matrix factorization, as described in [16]. To ensure a fair comparison, we limit the input matrix to a symmetric matrix.
- Fuzzy k-means (FKM) [32] applied at each layer: This approach is a variation of the popular k-means clustering algorithm. In traditional k-means, each data point is assigned to a single cluster. However, in this variant, data points can be assigned to multiple clusters with varying degrees of membership.

Next, we consider a simple undirected graph, such as the one depicted in Figure 1, to form our dataset. Specifically, we choose the parameters $L = 2$, $r_1 = 4$ and $r_2 = 2$. In this noise-free graph, there are two disjoint subgraphs of equal size. Each subgraph comprises two networks of the same size, with both networks sharing a common set of nodes. For instance, when considering $n = 18$, $n^* = 1$, we obtain the scenario illustrated in Figure 1. To add some randomness to the data, we incorporate symmetric white noise into the noiseless adjacency $\bar{\mathbf{X}}$. This is carried out by adding a small value ϵ as the standard deviation. By doing so, we create a noisy data matrix.

$$\mathbf{X} = \max(0, \bar{\mathbf{X}} + \epsilon \|\bar{\mathbf{X}}\|_F \frac{\mathbf{N}}{\|\mathbf{N}\|_F}),$$

where matrix $\mathbf{N}$ is symmetric, and its elements are randomly generated from a standard normal distribution.

We generated 10 random noise matrices, denoted for different levels of noise ϵ and combinations of n and n^* . These matrices were used in running DSSNMF and three other comparison algorithms (DSNMF, MulNMF, and FKM) with different values of p . To assess the results, we employed the mean removed spectral angle (MRSA) as the performance metric. The MRSA measures the average angular difference between the columns of the ground truth and computed factor matrices U_l , which are aligned to minimize differences. This calculation was performed over t runs. The MRSA between two vectors $\mathbf{x}$ and $\mathbf{y}$ can be calculated as follows:

$$MRSA(\mathbf{x}, \mathbf{y}) = \frac{100}{\pi} \arccos\left(\frac{\langle \mathbf{x} - \bar{\mathbf{x}}, \mathbf{y} - \bar{\mathbf{y}} \rangle}{\|\mathbf{x} - \bar{\mathbf{x}}\|_2 \|\mathbf{y} - \bar{\mathbf{y}}\|_2}\right).$$

We're easy to obtain $0 \leq MRSA(\mathbf{x}, \mathbf{y}) \leq 100$.

In Tables 1 and 2, we investigate the impact of varying the value of p on different combinations of n and n^* while maintaining a fixed noise level (ϵ) of 0.01, 0.05, 0.1, and 0.4, respectively. The results demonstrate that the MRSA values remain consistently stable within the range of p between 1.1 and 1.9. This finding suggests that our DSSNMF algorithm is robust to noise when the parameter β is appropriately fixed. Additionally, Table 3 examines the effect of different β values on the MRSA values under the same noise level. The findings reveal a continuous increase in the MRSA value as the range of β expands from 0.001 to 10.

Table 1. The effect of varying the value of p on the MRSA values at different noise levels when $n = 100$ and $n^* = 10$, fixing $\beta = 0.02$.

The first layer					
$(n, n^*, \epsilon) \setminus p$	$p = 1.1$	$p = 1.3$	$p = 1.5$	$p = 1.7$	$p = 1.9$
(100, 10, 0.01)	0.0521 ± 0.0028	0.0517 ± 0.0029	0.0519 ± 0.0028	0.0519 ± 0.0028	0.0519 ± 0.0028
(100, 10, 0.05)	0.2639 ± 0.0203	0.2637 ± 0.0195	0.2609 ± 0.0152	0.2615 ± 0.0156	0.2643 ± 0.0166
(100, 10, 0.1)	0.5440 ± 0.0335	0.5387 ± 0.0402	0.5340 ± 0.0338	0.5325 ± 0.0372	0.5355 ± 0.0428
(100, 10, 0.4)	2.7410 ± 0.0988	2.6969 ± 0.1023	2.6784 ± 0.1141	2.7084 ± 0.1241	2.7384 ± 0.1200
The second layer					
$(n, n^*, \epsilon) \setminus p$	$p = 1.1$	$p = 1.3$	$p = 1.5$	$p = 1.7$	$p = 1.9$
(100, 10, 0.01)	0.0472 ± 0.0046	0.0473 ± 0.0045	0.0475 ± 0.0047	0.0479 ± 0.0043	0.0521 ± 0.0176
(100, 10, 0.05)	0.3096 ± 0.0803	0.2978 ± 0.0579	0.2873 ± 0.0474	0.2881 ± 0.0480	0.3123 ± 0.0571
(100, 10, 0.1)	0.5628 ± 0.1115	0.5660 ± 0.1094	0.5827 ± 0.1445	0.6472 ± 0.2283	0.6099 ± 0.1979
(100, 10, 0.4)	2.8609 ± 0.7563	2.8609 ± 0.7563	2.8609 ± 0.7563	2.8609 ± 0.7563	2.8609 ± 0.7563

Table 2. The effect of varying the value of p on the MRSA values at different noise levels when $n = 18$ and $n^* = 1$, fixing $\beta = 0.02$.

The first layer				
$(n, n^*, \epsilon) \setminus p$	$p = 1.2$	$p = 1.4$	$p = 1.6$	$p = 1.8$
(18, 1, 0.01)	0.1069 ± 0.0129	0.1043 ± 0.0132	0.1062 ± 0.0144	0.1049 ± 0.0131
(18, 1, 0.05)	0.4734 ± 0.0578	0.4734 ± 0.0613	0.4758 ± 0.0627	0.4767 ± 0.0620
(18, 1, 0.1)	1.0336 ± 0.1284	1.0330 ± 0.1280	1.0312 ± 0.1277	1.0338 ± 0.1261
(18, 1, 0.4)	4.3235 ± 0.4812	4.3094 ± 0.4916	4.3201 ± 0.4911	4.3187 ± 0.4845
The second layer				
$(n, n^*, \epsilon) \setminus p$	$p = 1.2$	$p = 1.4$	$p = 1.6$	$p = 1.8$
(18, 1, 0.01)	1.5578 ± 6.3531	1.5455 ± 6.3113	1.5619 ± 6.3453	1.5616 ± 6.3286
(18, 1, 0.05)	5.9477 ± 9.1888	5.1045 ± 8.4058	5.9908 ± 9.3166	5.9533 ± 8.9989
(18, 1, 0.1)	4.8247 ± 8.8123	4.8180 ± 8.7312	4.8381 ± 8.8009	5.0625 ± 8.8486
(18, 1, 0.4)	20.7649 ± 14.5695	20.931 ± 13.5458	21.2376 ± 13.9992	21.3150 ± 13.5245

Table 3. The effect of varying the value of β on the MRSA values at different noise levels when $n = 50$ and $n^* = 5$, fixing $p = 1.4$.

The first layer						
$(n, n^*, \epsilon) \setminus \beta$	$\beta = 0.001$	$\beta = 0.01$	$\beta = 0.1$	$\beta = 1$	$\beta = 10$	$\beta = 100$
(50, 5, 0.01)	0.0709 ± 0.0037	0.0713 ± 0.0038	0.0722 ± 0.0038	0.0725 ± 0.0040	0.1119 ± 0.1194	0.2001 ± 0.4748
(50, 5, 0.05)	0.3615 ± 0.0232	0.3618 ± 0.0236	0.3620 ± 0.0232	0.3734 ± 0.0259	0.5007 ± 0.1606	0.6832 ± 0.5689
(50, 5, 0.1)	0.7388 ± 0.0459	0.7374 ± 0.0471	0.7369 ± 0.0478	0.7409 ± 0.0449	0.8479 ± 0.1669	1.5642 ± 2.0583
(50, 5, 0.4)	3.1757 ± 0.1779	3.1794 ± 0.1811	3.1932 ± 0.1895	3.3092 ± 0.1837	3.4730 ± 0.1858	3.3764 ± 0.2636
The second layer						
$(n, n^*, \epsilon) \setminus \beta$	$\beta = 0.001$	$\beta = 0.01$	$\beta = 0.1$	$\beta = 1$	$\beta = 10$	$\beta = 100$
(50, 5, 0.01)	0.0642 ± 0.0062	0.0686 ± 0.0214	0.0768 ± 0.0599	0.0875 ± 0.0824	0.2264 ± 0.2870	0.4803 ± 0.6225
(50, 5, 0.05)	0.3431 ± 0.0481	0.3228 ± 0.0481	0.3425 ± 0.0485	0.4626 ± 0.1691	0.5800 ± 0.4881	2.3267 ± 4.3214
(50, 5, 0.1)	0.7351 ± 0.1828	0.6850 ± 0.0907	0.6934 ± 0.0901	0.7068 ± 0.1140	1.3429 ± 1.1559	3.2424 ± 4.6656
(50, 5, 0.4)	3.6223 ± 1.0487	3.3539 ± 1.0101	3.4184 ± 0.7779	3.3132 ± 0.8888	3.7074 ± 0.8393	5.9160 ± 7.1708

In Tables 4 and 5, the experimental evaluation of the decomposition algorithm is presented, with the primary objective of assessing its performance. The quality of the decomposition is measured using the MRSA metric, which allows for an objective evaluation. Furthermore, a comparative analysis with other well-known algorithms is included,

adding depth to the assessment. Overall, these comprehensive findings contribute to a better understanding of the effectiveness and reliability of decomposition algorithms in this context. The numerical results of each visualization are shown in Figure 2.

Table 4. The MRSA of DSSNMF, DSNMF, MSNMF, and FKM on synthetic data was evaluated over 20 runs. The average and standard deviation of MRSA were compared with varying noise levels ϵ and network configurations, while keeping $r_1 = 4$, $\beta = 0.02$ and $p = 1.4$ at the first layer.

$(n, n^*, \epsilon) \backslash \text{Methods}$	DSSNMF	DSNMF	MulNMF	FKM
(18, 1, 0.01)	0.1043 ± 0.0132	0.1038 ± 0.0131	0.1045 ± 0.0138	0.5239 ± 0.0160
(18, 1, 0.05)	0.4734 ± 0.0613	0.4739 ± 0.0592	0.4727 ± 0.0594	0.7323 ± 0.0905
(18, 1, 0.1)	1.0330 ± 0.1280	1.0329 ± 0.1276	1.0276 ± 0.1204	1.2282 ± 0.1601
(18, 1, 0.4)	4.3049 ± 0.4916	4.3054 ± 0.4850	4.0792 ± 0.5227	4.5117 ± 0.5069
(50, 5, 0.01)	0.0713 ± 0.0038	0.0709 ± 0.0037	0.0662 ± 0.0041	1.1242 ± 0.0068
(50, 5, 0.05)	0.3612 ± 0.0233	0.3614 ± 0.0231	0.3359 ± 0.0250	1.1812 ± 0.0280
(50, 5, 0.1)	0.7370 ± 0.0471	0.7375 ± 0.0464	0.6682 ± 0.0469	1.3218 ± 0.0625
(50, 5, 0.4)	3.1759 ± 0.1810	3.1757 ± 0.1779	2.7132 ± 0.1786	3.1948 ± 0.1565
(100, 10, 0.01)	0.0516 ± 0.0030	0.0522 ± 0.0028	0.0451 ± 0.0024	1.1249 ± 0.0033
(100, 10, 0.05)	0.2602 ± 0.0154	0.2611 ± 0.0151	0.2301 ± 0.0112	1.1482 ± 0.0263
(100, 10, 0.1)	0.5317 ± 0.0355	0.5441 ± 0.0335	0.4642 ± 0.0257	1.2321 ± 0.0435
(100, 10, 0.4)	2.6838 ± 0.1125	2.7448 ± 0.1010	1.9678 ± 0.0763	2.4737 ± 0.1056

Table 5. The MRSA of DSSNMF, DSNMF, MSNMF, and FKM on synthetic data was evaluated over 20 runs. The average and standard deviation of MRSA were compared with varying noise levels ϵ and network configurations, while keeping $r_2 = 2$, $\beta = 0.02$ and $p = 1.4$ at the second layer.

$(n, n^*, \epsilon) \backslash \text{Methods}$	DSSNMF	DSNMF	MulNMF	FKM
(18, 1, 0.01)	1.5455 ± 6.3113	1.5700 ± 6.3411	2.6463 ± 6.2355	0.1949 ± 0.0639
(18, 1, 0.05)	5.1045 ± 8.4058	5.9290 ± 9.1747	9.9505 ± 10.0311	0.9365 ± 0.2825
(18, 1, 0.1)	4.8180 ± 8.7312	4.8669 ± 8.8282	20.4606 ± 12.5311	2.0169 ± 0.6467
(18, 1, 0.4)	20.932 ± 13.5458	21.1335 ± 14.5723	30.7843 ± 11.2594	11.0709 ± 5.7227
(50, 5, 0.01)	0.0638 ± 0.0057	0.0645 ± 0.0063	8.3818 ± 3.9566	0.0712 ± 0.0074
(50, 5, 0.05)	0.3208 ± 0.0490	0.3434 ± 0.0479	0.8620 ± 0.4740	0.3596 ± 0.0317
(50, 5, 0.1)	0.6812 ± 0.0917	0.7218 ± 0.1217	7.1485 ± 4.0679	0.7038 ± 0.0856
(50, 5, 0.4)	3.3458 ± 0.9142	3.6222 ± 1.0487	25.3546 ± 11.2502	3.1399 ± 0.4790
(100, 10, 0.01)	0.0464 ± 0.0043	0.0472 ± 0.0047	7.6693 ± 4.8342	0.0504 ± 0.0048
(100, 10, 0.05)	0.2880 ± 0.0472	0.2966 ± 0.0522	2.2166 ± 1.2100	0.2506 ± 0.0142
(100, 10, 0.1)	0.5527 ± 0.1748	0.5630 ± 0.1113	1.5343 ± 1.0768	0.5023 ± 0.0317
(100, 10, 0.4)	2.8309 ± 0.7563	2.8612 ± 0.7551	26.3690 ± 10.2289	2.1006 ± 0.1639

Transitioning to Table 4, we evaluate the MRSA of the first layer factorization, comparing the performance of DSSNMF, DSNMF, MulNMF, and FKM. Notably, DSSNMF, DSNMF, and MulNMF outperform FKM, illustrating the effectiveness of our proposed algorithm. The results obtained from these three methods are comparable, with MulNMF expected to perform better theoretically due to its independent optimization of the first layer, a result that is confirmed by our experimental findings. However, even though DSSNMF and DSNMF also consider the factorization of the second layer, both methods demonstrate promising experimental results, with DSSNMF exhibiting a slight advantage over DSNMF. Additionally, DSSNMF offers the benefit of further enhancing the MRSA by leveraging the parameter β .

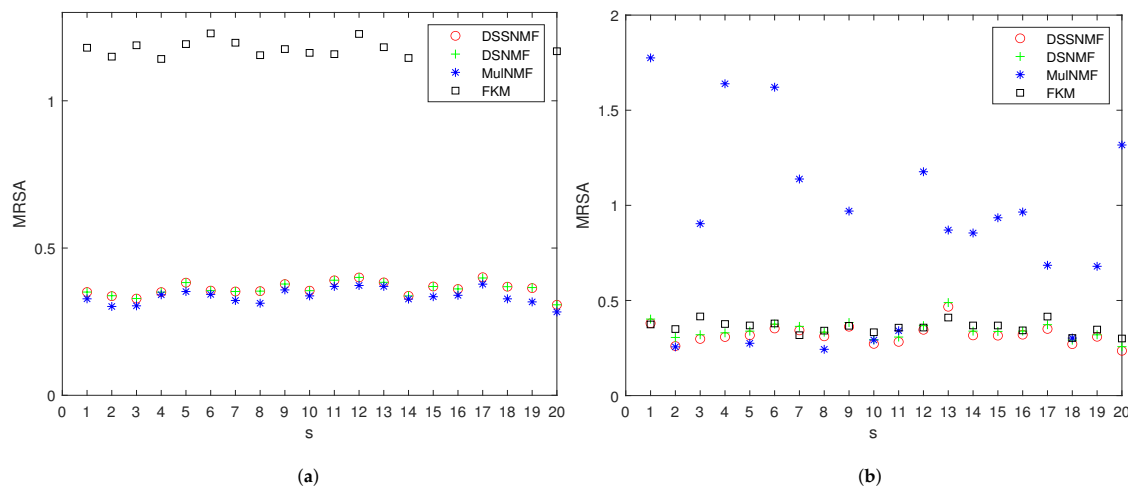


Figure 2. Take $(n, n^*, \epsilon) = (50, 5, 0.05)$ as an example. MRSA visualization plots of DSSNMF and the three comparison algorithms run 20 times on synthetic data when $r_1 = 4$, $r_2 = 2$, $\beta = 0.02$ and $p = 1.4$. The numerical results are shown in Figure 1. The x-axis of Figure 2 refers to the s -th experiments. (a) The first layer of factorization. (b) The second layer of factorization.

Surprisingly, as shown in Table 5, which presents the results of the second layer factorization, DSSNMF and DSNMF excel at extracting internal features and capturing the hierarchical structure through deep factorization techniques, surpassing other algorithms. In contrast, MuNMF performs poorly, exposing the limitations of standalone layer-by-layer factorization. On the other hand, FKM achieves the best results in MRSA. However, it is important to acknowledge that FKM solely operates on the original data matrix, disregarding the hierarchical information embedded within the dataset. In contrast, as the number of layers of decomposition increases, DSSNMF outperforms DSNMF in extracting internal features and capturing the internal hierarchical structure of the dataset.

Overall, the results from these tables emphasize the effectiveness of our proposed DSSNMF algorithm in achieving robustness against noise, superior performance in comparison to existing methods, and its ability to uncover the hierarchical nature of the dataset.

4.2. Post-Traumatic Stress Disorder (PTSD)

Network analysis has recently gained traction in computational psychiatry and psychology, particularly in studying mood, attitudes, and psychological disorders. The existing literature in psychology offers explanations for certain networks [33,34]. Additionally, researchers have extensively discussed biased correlation and partial correlation networks as effective models for estimating psychological datasets. In our study, we work with an original data matrix that consists of ratings from m subjects on n symptoms denoted as $\mathbf{Y} \in \mathbb{R}_+^{m \times n}$. Here, the number of rows represents the number of patients, while the number of columns represents the number of symptoms. Consequently, the matrix $\mathbf{Y}$ captures the assessment data for m patients with regard to n symptoms. The matrix of partial correlation coefficients for the n patients is $\mathbf{X} \in \mathbb{R}_+^{n \times n}$ and its elements are calculated as $\mathbf{X}(i, j) = \frac{\mathbf{K}(i, j)}{\sqrt{\mathbf{K}(i, i)\mathbf{K}(j, j)}}$, here, $\mathbf{K}$ denotes the inverse of the variance–covariance matrix Σ , i.e., $\mathbf{K} = \Sigma^{-1}$. For a detailed explanation of this calculation and its underlying principles, please refer to [34].

Considering a graph represented by the adjacency matrix $\mathbf{X}$, where each node is a symptom, it is important to identify the community to which each symptom belongs. This analysis would enable us to evaluate how symptoms respond to interventions like medication and examine the interconnectedness among the symptoms. However, existing research in this area has focused on extracting only one layer of disjoint communities, limiting our understanding of the intricate interactions within the network. To address

this limitation, we aim to explore multiple layers of community structure to gain a more comprehensive understanding of symptom interactions.

Next, we examined a dataset of 359 women with post-traumatic stress disorder (PTSD) as assessed by the PTSD Symptom Self Rating Scale (PSS-SR) [35,36]. The scale consists of 17 items that diagnose PTSD according to the DSM-IV criteria [37] (the fifth edition is recorded as DSM-5 [38]) and measures the severity of PTSD symptoms. This scale is widely used in the field of mental health to classify and assess various mental disorders. The 17 symptoms are categorized into three clusters: the Reexperiencing cluster (including Intrusive thoughts, Nightmares, Flashbacks, and Emotionally upset), the Avoidance cluster (including Avoid thoughts and feelings, Avoid places and activities, Psychogenic amnesia, Loss of interest, Detached from others, Restricted affect, and Foreshortened sense of future), and the Arousal cluster (including Sleep disturbance, Irritability, Difficulty concentrating, Hyperalertness, Increased startle, and Physical reactivity). Each item corresponds to the frequency of specific behaviors considered characteristic of the pathology.

The symptom network for the PTSD dataset was constructed in R, and regularized estimation of a partial correlation network was performed using extended bayesian information criterion (EBIC) [39] selection. The full R code for this analysis can be found in the supplementary material to the literature [36]. The visualized symptom network is shown in Figure 3.

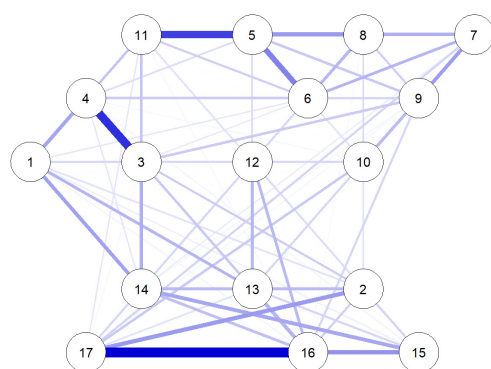


Figure 3. The symptom network for the PTSD dataset. This network containing the 17 DSM-IV symptoms of PTSD, and the thickness and brightness of an edge indicate the association strength.

The DSSNMF algorithm was initialized using the LM method and applied to the symptom network. The algorithm successfully extracted three layers of communities from the network. In the first layer, four communities were identified, followed by three communities in the second layer, and two communities in the third layer. The parameter values used for DSSNMF were as follows: $L = 3$, $r_1 = 4$, $r_2 = 3$, and $r_3 = 2$. To demonstrate the effectiveness of the algorithm, Table 6 presents the three-layer communities extracted by DSSNMF when the parameter β is set to 0.05. Similarly, Table 7 shows the three-layer communities obtained when the parameter β is set to 0.5. For the sake of simplicity, each node was assigned to a major cluster, thereby facilitating the understanding of how symptoms are grouped within the PTSD dataset. It is important to note that each node could belong to one or more communities based on its connections and interactions within the network.

Table 6. Hierarchical communities extracted by DSSNMF when $\beta = 0.05$.

$\beta = 0.05$	First Layer	Second Layer	Third Layer
C1	{1, 3, 4, 13, 14}	{1, 3, 4}	{3, 4, 5, 6, 7, 8, 9, 10, 11}
C2	{2, 10, 12, 13, 14, 15, 16, 17}	{1, 2, 10, 12, 13, 14, 15, 16, 17}	{1, 2, 3, 10, 12, 13, 14, 15, 16, 17}
C3	{5, 6, 7, 8, 9, 10, 11, 12, 13}	{5, 6, 7, 8, 9, 10, 11}	
C4	{12, 13, 14, 17}		

Table 7. Hierarchical communities extracted by DSSNMF when $\beta = 0.5$.

$\beta = 0.5$	First Layer	Second Layer	Third Layer
C1	{1, 3, 4, 13, 14}	{1, 3, 4}	{3, 4, 5, 6, 7, 8, 9, 10, 11}
C2	{2, 12, 13, 14, 15, 16, 17}	{1, 2, 3, 10, 12, 13, 14, 15, 16, 17}	{1, 2, 3, 10, 12, 13, 14, 15, 16, 17}
C3	{4, 5, 6, 7, 8, 9, 10, 11, 12, 13}	{1, 3, 4, 5, 6, 7, 8, 9, 10, 11}	
C4	{12, 13, 14}		

The assignment of nodes to communities in the DSSNMF algorithm is based on the membership degrees of the nodes in the factor matrices. The largest membership degree determines the initial assignment of a node to a community. Subsequently, if the membership degree of a node in a new community is at least 60% of its membership degree in the latest assigned community, it is assigned to the new community as well.

By observing Tables 6 and 7, it can be seen that changing the parameter β leads to changes in the communities to which some nodes belong. For example, node 4 in the first layer of the community changed when the parameter β changed from 0.05 to 0.5. This indicates that the parameter β affects the proportions of this node in the factor matrix.

These changes provide preliminary insights into the effect of this symptom (Take node 4 as an example) on other parts of the network. However, further analysis and interpretation are required to fully understand the implications of these changes and their significance in relation to the overall network structure.

The changes in the community assignments of nodes when varying the parameter β provide insights into the underlying structure and relationships within the network. By observing the shifting memberships of nodes across different communities, we can discern how certain symptoms are related to or interact with others.

For example, in Table 6, when $\beta = 0.05$, node 12 appears in communities C2, C3, and C4, while node 13 is present in communities C1, C2, C3, and C4 in the first layer. This suggests that these symptoms (represented by nodes 12 and 13) have connections and overlaps with multiple symptom clusters. The change in the community assignments indicates that they may play a role in different aspects of the disorder or have varied associations with other symptoms depending on the value of β .

These findings highlight the complex nature of the symptom network and provide preliminary insights into the potential interactions and relationships between symptoms. By examining the hierarchical information of the community structure, we gain a deeper understanding of how symptoms are grouped together and how they relate to each other within the PTSD dataset.

Indeed, the ability of DSSNMF to classify symptoms into distinct clusters and extract symptoms that are closely related to multiple clusters is valuable for understanding the interplay and relationships between different symptom clusters within the PTSD dataset. This information can provide insights into the underlying mechanisms and dynamics of the disorder, helping researchers and clinicians better understand the complex interactions between symptoms and potentially guide more targeted interventions and treatment approaches.

Furthermore, the extracted communities align with the transition from DSM-IV-TR to DSM-IV, demonstrating the consistency and interpretability of the hierarchical information obtained through the DSSNMF algorithm. This indicates the practical significance of the algorithm in providing meaningful clinical interpretations and potential applications in the psychological domain.

Overall, the application of the DSSNMF algorithm in this study contributes to the field of deep NMF and provides valuable insights into the hierarchical structure of datasets generated in fields such as network analysis and psychology. Further research can explore the potential of hierarchical community extraction in other domains, which presents a fascinating avenue for future investigations.

5. Conclusions

Deep matrix factorization (deep MF) methods have gained significant attention in recent years due to their capability to hierarchically extract human understandable features. When dealing with high-dimensional and noisy data, data analysis and processing can become a challenging task, making methods that enhance interpretability highly appreciated. Therefore, this paper aims to enhance the deep MF algorithm's capability to extract information about the intrinsic hierarchical structure of datasets generated in fields such as network analysis and psychology, where the data matrices are symmetric non-negative matrices. To achieve this, we propose a deep L_p smooth symmetric non-negative matrix factorization (DSSNMF) algorithm. Experimental results on the PTSD dataset and the synthetic dataset for network analysis demonstrate that the DSSNMF algorithm successfully achieves this goal. The L_p smoothing condition imposed by the algorithm improves the noise immunity of the algorithm. Furthermore, we extracted non-disjoint communities in a network of psychiatric symptoms in PTSD, which resulted in meaningful clinical interpretations. Exploring the potential of hierarchical community extraction in the psychological domain could be a fascinating avenue for future research.

Author Contributions: Conceptualization, S.L. and C.S.; methodology, S.L. and L.L.; software, C.S. and Z.C.; writing—original draft preparation, S.L.; writing—review and editing, C.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by the National Natural Science Foundation of China under Grant 12161020, 12061025, and partially funded by the Natural Science Foundation of Educational Commission of Guizhou Province under Grant Qian-Jiao-He KY Zi [2021]298, Guizhou Provincial Basis Research Program (Natural Science) (QKHJC-ZK[2023]YB245, QKHJC [2020]1Z002).

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: The authors are grateful to the editor and reviewers for their valuable review comments on our work. In addition, we would like to express our special thanks to the Science and Technology Department of Guizhou Province for listing the research topic of this thesis, “Application Research of Deep Nonnegative Matrix Factorization in Hyperspectral Analysis and Image Recognition”, as a Guizhou Provincial Basis Research Program (Natural Science) starting from 2024 (Project Announcement No. 627 in the list, Applicant and Director: Shunli Li).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Lee, D.; Seung, H.S. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*; MIT Press: Cambridge, MA, USA, 2000.
2. Xu, W.; Gong, X.L.; Gong, Y. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Toronto, ON, Canada, 28 July–1 August 2003; ACM: New York, NY, USA, 2003; pp. 267–273.
3. Shahnaz, F.; Berry, M.W.; Pauca, V.P.; Plemmons, R.J. Document clustering using nonnegative matrix factorization. *Inf. Process. Manage.* **2006**, *42*, 373–386. [\[CrossRef\]](#)
4. Cai, D.; He, X.; Han, J.; Huang, T.S. Graph regularized nonnegative matrix factorization for data representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *33*, 1548–1560. [\[PubMed\]](#)
5. Pascual-Montano, A.; Carazo, J.M.; Kochi, K.; Lehmann, D.; Pascual-Marqui, R.D. Nonsmooth nonnegative matrix factorization (nsnmf). *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *28*, 403–415. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Fu, X.; Huang, K.; Sidiropoulos, N.D.; Ma, W.K. Nonnegative matrix factorization for signal and data analytics: Identifiability, algorithms, and applications. *IEEE Signal. Proc. Mag.* **2019**, *36*, 59–80. [\[CrossRef\]](#)
7. Vaswani, N.; Bouwmans, T.; Javed, S.; Narayanamurthy, P. Robust subspace learning: Robust PCA, robust subspace tracking, and robust subspace recovery. *IEEE Signal. Proc. Mag.* **2018**, *35*, 32–55. [\[CrossRef\]](#)
8. Liu, Z.; Yuan, G.; Luo, X. Symmetry and nonnegativity-constrained matrix factorization for community detection. *IEEE/CAA J. Autom. Sin.* **2022**, *9*, 1691–1693. [\[CrossRef\]](#)
9. Shi, X.; Lu, H.; He, Y.; He, S. Community detection in social network with pairwise constrained symmetric non-negative matrix factorization. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Paris, France, 25–28 August 2015; pp. 541–546.

10. Li, S.; Lu, L.; Q.; Chen, Z. Graph-regularized, sparsity-constrained non-negative matrix factorization with earth mover's distance metric. *Mathematics* **2023**, *11*, 1894. [\[CrossRef\]](#)
11. Pan, J.; Ng, M.K.; Liu, Y.; Zhang, X.; Yan, H. Orthogonal nonnegative Tucker decomposition. *SIAM J. Sci. Comput.* **2021**, *43*, 55–81. [\[CrossRef\]](#)
12. Kuang, D.; Ding, C.; Park, H. Symmetric nonnegative matrix factorization for graph clustering. In Proceedings of the 2012 SIAM International Conference on Data Mining, Anaheim, CA, USA, 26–28 April 2012; pp. 106–117.
13. Huang, K.; Fu, X.; Sidiropoulos, N.D. Anchor-free correlated topic modeling: Identifiability and algorithm. In Proceedings of the Advances in Neural Information Processing Systems 29, Barcelona, Spain, 5–10 December 2016.
14. Wang, J.; Zhang, X.L. Deep NMF topic modeling. *Neurocomputing* **2023**, *515*, 157–173. [\[CrossRef\]](#)
15. Cichocki, A.; Zdunek, R. Multilayer nonnegative matrix factorisation. *Electron. Lett.-IEE* **2006**, *42*, 947. [\[CrossRef\]](#)
16. Cichocki, A.; Zdunek, R. Multilayer nonnegative matrix factorization using projected gradient approaches. *Int. J. Neural Syst.* **2007**, *17*, 431–446. [\[CrossRef\]](#)
17. Zhao, H.; Ding, Z.; Fu, Y. Multi-view clustering via deep matrix factorization. *Mathematics* **2017**, *31*. [\[CrossRef\]](#)
18. Trigeorgis, G.; Bousmalis, K.; Zafeiriou, S.; Schuller, B.W. A deep matrix factorization method for learning attribute representations. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 417–429. [\[CrossRef\]](#)
19. Chen, W.-S.; Zeng, Q.; Pan, B. A survey of deep nonnegative matrix factorization. *Neurocomputing* **2022**, *491*, 305–320. [\[CrossRef\]](#)
20. De Handschutter, P.; Gillis, N. A consistent and flexible framework for deep matrix factorizations. *Pattern Recognit.* **2023**, *134*, 109102. [\[CrossRef\]](#)
21. Smith, E.A.J. Hierarchical feature extraction through deep matrix factorization. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 2356–2369.
22. Xue, H.-J.; Dai, X.; Zhang, J.; Huang, S.; Chen, J. Deep matrix factorization models for recommender systems. *IJCAI* **2017**, *17*, 3203–3209.
23. Hajiveisheh, A.; Seyedi, S.A.; Tab, F.A. Deep asymmetric nonnegative matrix factorization for graph clustering. *Pattern Recognit.* **2024**, *148*, 110179. [\[CrossRef\]](#)
24. Li, X.; Zhu, Z.; Li, Q.; Liu, K. A provable splitting approach for symmetric nonnegative matrix factorization. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 2206–2219. [\[CrossRef\]](#)
25. De Handschutter, P.; Gillis, N.; Blekic, W. Deep symmetric matrix factorization. *Proc. IEEE* **2023**, 635–639.
26. Nesterov, Y.E. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. *Dokl. Akad. Nauk SSSR* **1983**, *269*, 543–547.
27. Blondel, V.D.; Guillaume, J.-L.; Lambiotte, R.; Lefebvre, E. Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**, *2008*, 10008. [\[CrossRef\]](#)
28. Luo, X.; Liu, Z.; Jin, L.; Zhou, Y.; Zhou, M. Symmetric nonnegative matrix factorization-based community detection models and their convergence analysis. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 1203–1215. [\[CrossRef\]](#) [\[PubMed\]](#)
29. Luo, X.; Shang, M. Symmetric non-negative latent factor models for undirected large networks. *IJCAI* **2017**, 2435–2442.
30. Feng, X.-R.; Li, H.-C.; Li, J.; Du, Q.; Plaza, A.; Emery, W.J. Hyperspectral unmixing using sparsity-constrained deep nonnegative matrix factorization with total variation. *IEEE Trans. Geosci. Remote. Sens.* **2018**, *56*, 6245–6257. [\[CrossRef\]](#)
31. Zhang, W.; Zhang, X.; Wang, H.; Chen, D. A deep variational matrix factorization method for recommendation on large scale sparse dataset. *Neurocomputing* **2019**, *334*, 206–218. [\[CrossRef\]](#)
32. Bezdek, J.C. *Pattern Recognition with Fuzzy Objective Function Algorithms*; Springer Science & Business Media: Berlin, Germany, 2013.
33. Costantini, G.; Epskamp, S.; Borsboom, D.; Perugini, M.; Mottus, R.; Waldorp, L.J.; Cramer, A.O. State of the art personality research: A tutorial on network analysis of personality data in R. *J. Res. Personal.* **2015**, *54*, 13–29. [\[CrossRef\]](#)
34. Epskamp, S.; Fried, E.I. A tutorial on regularized partial correlation networks. *Psychol. Methods* **2018**, *23*, 617. [\[CrossRef\]](#)
35. Foa, E.B.; Riggs, D.S.; Dancu, C.V.; Rothbaum, B.O. Reliability and validity of a brief instrument for assessing post-traumatic stress disorder. *J. Trauma. Stress* **1993**, *6*, 459–473.
36. Armour, C.; Fried, E.I.; Deserno, M.K.; Tsai, J.; Pietrzak, R.H. A network analysis of DSM-5 posttraumatic stress disorder symptoms and correlates in US military veterans. *J. Anxiety Disord.* **2017**, *45*, 49–59. [\[CrossRef\]](#)
37. Segal, D.L. Diagnostic and Statistical Manual of Mental Disorders (DSM-IV-TR). *Corsini Encycl. Psychol.* **2010**, 1–3.
38. American Psychiatric Association. *Diagnostic and Statistical Manual of Mental Disorders: DSM-5*, 5th ed.; American Psychiatric Association: Washington, DC, USA, 2013.
39. Chen, J.; Chen, Z. Extended Bayesian information criteria for model selection with large model spaces. *Biometrika* **2008**, *95*, 759–771. [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.