Graph-Regularized Non-Negative Tensor-Ring Decomposition for Multiway Representation Learning

Yuyuan Yu, Guoxu Zhou, Ning Zheng, Yuning Qiu, Shengli Xie, Fellow, IEEE, and Qibin Zhao, Senior Member, IEEE

Abstract—Tensor-ring (TR) decomposition is a powerful tool for exploiting the low-rank property of multiway data and has been demonstrated great potential in a variety of important applications. In this article, non-negative TR (NTR) decomposition and graph-regularized NTR (GNTR) decomposition are proposed. The former equips TR decomposition with the ability to learn the parts-based representation by imposing non-negativity on the core tensors, and the latter additionally introduces a graph regularization to the NTR model to capture manifold geometry information from tensor data. Both of the proposed models extend TR decomposition and can be served as powerful representation learning tools for non-negative multiway data. The optimization algorithms based on an accelerated proximal gradient are derived for NTR and GNTR. We also empirically justified that the proposed methods can provide more interpretable and physically meaningful representations. For example, they are able to extract parts-based components with meaningful color and line patterns from objects. Extensive experimental results demonstrated that the proposed methods have better performance than state-of-the-art tensor-based methods in clustering and classification tasks.

Index Terms—Non-negative tensor decomposition, representation learning, tensor learning.

I. INTRODUCTION

Extracting meaningful and interpretable low-dimensional representations from high-dimensional data is a fundamental task in the fields of data mining, signal processing, and machine learning. One of the main challenges is how to capture physically meaningful low-dimensional features from high-dimensional data while providing experiment-related interpretability. Non-negative matrix factorization (NMF) [1] has gained much attention and is used to approximate the data matrix by the multiplication of two non-negative factor matrices. The non-negativity of representation makes the components sparse and suppresses the energy of the small amount of noise in the data [2]. Therefore, NMF has been shown to be effective in feature extraction [3]–[5], multiview clustering [6], etc.

With the advancement of data acquisition technology, an increasing number of high-dimensional data have been collected. These data are also known as the multiway data or tensor data. Hazan et al. [15] argued that preserving the multiway structure of tensor data contributes to learn the parts-based components of image objects. Hence, they proposed non-negative tensor factorization (NTF) to approximate the grayscale image database using a linear sum of the outer products of non-negative vectors. The experimental results demonstrated that the NTF method can generate the parts-based sparse features. The NTF method often gives the most compact representation but suffer from a convergence issue. Kim and Choi [14] developed non-negative Tucker decomposition (NTD) to introduce a core tensor that represents the connections and interactions between different factor matrices to discover the most significant links between components and enhance the interpretability of the model. NTD has been
applied to image denoising [14] and blind source separation [17]. However, the size of the core tensor in NTD expands exponentially with respect to the order of data, thereby leading to the high complexity in terms of model calculation and estimation [17]. To overcome this shortcoming, Fonal and Zdunek [18] proposed hierarchical NTD (HNTD), at the cost of frequently searching a binary tree [19]. Later, a non-negative tensor train (NTT) was proposed [20], which represents an $N$th-order tensor as multilinear products of $N$ low-order core tensors. NTT is much easier for the implementation in practice as no binary tree needs to be determined like HNTD [19]. Moreover, NTT can alleviate the curse of dimensionality because its number of parameters is linear to the order of tensor data [21]. However, NTT has the following limitations.

1) The boundary rank is restricted to be exactly one, which often leads to a highly unbalanced structure, that is, the core tensors at the center are often significantly larger than the others [22].

2) The efficiency of NTT is also highly dependent on the permutation of tensor dimensions, and finding the optimal permutation is quite expensive [23].

In this article, we propose the non-negative TR (NTR) method for high-order non-negative tensor data analysis by introducing a local geometric information of tensor data, we develop the graph-regularized NTR (GNTR) method by introducing a graph regularization term to the NTR model. Specifically, the main contributions and significance of this article can be summarized as follows.

1) An NTR model is proposed which inherits the advantages of both non-negative representation and TR decomposition. Particularly, compared with existing non-negative tensor decomposition models, it gives a more compact and more balanced representation, free of the circular shift permutation issue of data, and easy to implement.

2) A GNTR model is proposed that inherits the advantages of the NTR model and captures the manifold geometrical structure of tensor data. Compared with existing graph-regularized non-negative tensor decomposition models, it combined geometrical structure information modeled by the nearest neighbor graph in the more compact and more balanced representation.

3) We develop an efficient iterative algorithm based on the accelerated proximal gradient method to efficiently optimize the NTR and GNTR models and prove its convergence property theoretically.

4) The experimental results demonstrate that the NTR and GNTR methods extract the parts-based components with meaningful color and line patterns to provide an interpretable result, and the GNTR method achieves better performance than the state-of-the-art methods in clustering and classification tasks.

The remainder of this article is organized as follows. In Section II, the notations and preliminaries are introduced. In Section III, the NTR and GNTR methods are developed. Finally, simulations on five public databases are presented in Section IV, followed by conclusions in Section V.

II. NOTATIONS AND PRELIMINARIES

A. Notations

We review the related definitions of tensor decomposition. The notations used in this article are presented in Table I.

Definition 1 (Inner Product): Given two tensors $X$ and $Y$ of the same size $I_1 \times I_2 \times \cdots \times I_N$, their inner product is defined as the sum of the products of their elements

$$\langle X, Y \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1 i_2 \cdots i_N} y_{i_1 i_2 \cdots i_N}.$$  

Definition 2 (Mode-n Product): Given a tensor $X \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and a matrix $U \in \mathbb{R}^{I_M \times I_N}$, their inner product is defined as $X \times_n U \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M \times \cdots \times I_N}$. Elementwise, we have the following equation:

$$(X \times_n U)_{i_1 \cdots i_{n-1} j_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \cdots i_{n-1} i_n j_{n+1} \cdots i_N}.$$  

Definition 3 (Mode-n Unfolding): Given a tensor $X \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, its mode-$n$ unfolding is obtained by fixing all the indices except $I_n$, which is denoted as $X_{(n)} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_{n-1} \times I_{n+1} \times \cdots \times I_N}$.

The classical mode-$n$ unfolding of $X$ is used in TR operations [23] and is defined as $X_{[n]} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_{n-1} \times I_{n+1} \times \cdots \times I_N}$.

In contrast to the mode-$n$ unfolding operation, the folding operation is an inverse operation of the mode-$n$ unfolding and is defined as follows:

$$fold(X_{(n)}) = X.$$

Definition 4 (Contracted Product): Given two tensors $X \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and $Y \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$, where $I_N = J_1$. The contracted product of $X$ and $Y$ is defined as follows:

$$Z = X \times_1 Y = X^1 Y$$

where $Z \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N \times I_J \times I_M}$, and its element can be obtained by the following equation:

$$z_{i_1, i_2, \ldots, i_{N-1}, j_2, j_3, \ldots, j_M} = \sum_{i_N=1}^{I_N} x_{i_1, i_2, \ldots, i_N} y_{i_N j_2, \ldots, j_M}.$$  

Definition 5 (Multilinear Product): Given two tensors $G^{(n)} \in \mathbb{R}^{R_{n} \times I_{1} \times R_{n+1}}$ and $G^{(n+1)} \in \mathbb{R}^{R_{n+1} \times I_{2} \times R_{n+2}}$, the multilinear product of $G^{(n)}$ and $G^{(n+1)}$ is defined as follows:

$$G^{(n,n+1)} = G^{(n)} \times G^{(n+1)}.$$  

| Table I: List of the Notation |
|-----------------------------|
| $X$ | A vector |
| $\mathcal{X}$ | A tensor |
| $\mathcal{Y}$ | A matrix |
| $T_n$ | Trace |
| $\| \cdot \|_F$ | Frobenius norm |
| $\circ$ | Outer product |
| $\otimes$ | Khatri–Rao product |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
where \( G^{(n,n+1)} \in \mathbb{R}^{R_n \times I_{n+1} \times R_{n+2}} \) and its element can be obtained by the following equation:

\[
G^{(n,n+1)}_{r_n, (i_n)_1+1, t_{n+1}, r_{n+2}} = \sum_{i_{n+1}=1}^{R_{n+1}} G^{(n)}_{r_n, r_{n+1}} G^{(n+1)}_{r_{n+1}, t_{n+1}, r_{n+2}}
\]

where \( r_n = 1, 2, \ldots, R_n \) and \( i_n = 1, 2, \ldots, I_n \).

To facilitate the comparisons, Fig. 1 depicts the graphical representation of the basic symbols and tensor operations via a tensor network diagram [24].

**B. NTF**

Hazan et al. [15] proposed the NTF model based on the CP structure and proved its effectiveness in image representation. Given a non-negative tensor \( \mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \), it can be represented as the sum of \( R \) rank-one non-negative tensors by using NTF as follows:

\[
\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \otimes \mathbf{a}_r^{(2)} \otimes \cdots \otimes \mathbf{a}_r^{(N)}
\]

where \( R \) denotes the non-negative rank of NTF. As presented in [25], NTF treats the rank of tensor data with different modes as the same, which is contrary to some practical applications.

**C. NTD**

The NTD model is another popular non-negative tensor decomposition model [14]. Given a non-negative tensor \( \mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \), it can be approximated by using NTD as follows:

\[
\mathcal{X} = \mathcal{G} \times_1 A^{(1)} \times_2 A^{(2)} \cdots \times_N A^{(N)}
\]

where \( \mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N} \) denotes the non-negative core tensor, \( A^{(n)} \in \mathbb{R}^{I_1 \times R_n}, n = 1, 2, \ldots, N \) denote the non-negative factor matrices, and \( R = [R_1, R_2, \ldots, R_N] \) denotes the TR rank. Each element of \( \mathcal{X} \) can be written by TR as follows:

\[
x_{i_1, i_2, \ldots, i_N} = \text{Tr} \left( \prod_{n=1}^{N} G_n(i_n) \right)
\]

where \( G_n(i_n) \in \mathbb{R}^{R_n \times R_{n+1}} \) is the \( i_n \)th lateral slice matrix of the \( n \)th core tensor \( G^{(n)} \). The TR model satisfies the invariance of the circular translation [23]. Therefore, the core tensors of TR can be treated more balanced. Compared with the TT model, the balanced structure makes the TR model more suitable for complex data analysis. Fig. 2 shows the tensor network diagrams of the tensor models with different structures.

**D. NTT**

Lee et al. [20] developed the NTT model based on the TT structure and verified that NTT combined with the Tucker (NTT-Tucker) model achieves high clustering performance with lower storage costs than NTD for image representation. Given a non-negative tensor \( \mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \), it can be expressed as the contracted product of a series of non-negative low-order core tensors by using NTT as follows:

\[
\mathcal{X} = Z^{(1)} \times_1 Z^{(2)} \times_1 \cdots \times_1 Z^{(N)}
\]

where \( Z^{(n)} \in \mathbb{R}^{R_n \times I_n \times R_{n+1}} \), \( n = 1, 2, \ldots, N \) denote the non-negative core tensors. \( R = [R_1, R_2, \ldots, R_N] \) denotes the non-negative NTT rank. There is an additional rank constraint for NTT, which is \( R_1 = \sum_{n=1}^{N} R_n = I_{N+1} \).

NTT is suitable for the high-order tensor representation because it can alleviate the curse of dimensionality. However, the rank constraint is lead to NTT limited the representation ability and flexibility.

**E. TR**

The TR model [21] has recently received attention and has been shown to be powerful and efficient in various applications, e.g., image completion [26], hyperspectral image denoising [27], deep multimodal feature fusion [28], and neural network compression [29]. Given a tensor \( \mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \), it can be represented as circular contractions over a sequence of the 3rd-order core tensors by TR as follows:

\[
\mathcal{X} = \text{TR} \left( G^{(1)}, G^{(2)}, \ldots, G^{(N)} \right)
\]
III. GRAPH-REGULARIZED NON-NEGATIVE TENSOR-RING DECOMPOSITION

A. Objective Function of NTR

Previous works [26]–[29] demonstrated that the TR model is a powerful tool for high-order tensor data analysis. However, the TR model cannot learn the parts-based components of non-negative tensor objects because its representation is mixed signs. In this section, we develop the NTR model to overcome this problem. As we assume Gaussian noise on the non-negative tensor data \( X \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \), the Frobenius norm is naturally applied in our model to quantify the difference between data and non-negative representation.

\[
\min_{G^{(1)}, \ldots, G^{(N)}} \frac{1}{2} \left\| X - \text{NTR} \left( G^{(1)}, G^{(2)}, \ldots, G^{(N)} \right) \right\|_F^2 \\
\text{s.t.} \quad G^{(n)} \geq 0, \quad n = 1, 2, \ldots, N
\]  

(3)

where \( G^{(n)} \in \mathbb{R}_{+}^{R_n \times I_1 \times \ldots \times I_n} \) for \( n = 1, 2, \ldots, N \) denote the non-negative core tensors and \( R = [R_1, R_2, \ldots, R_N] \) with \( R_1 = R_N + 1 \) is the non-negative TR rank. NTR(\( G^{(1)}, G^{(2)}, \ldots, G^{(N)} \)) is defined as the circular contraction of the non-negative core tensors. Specifically, the classical mode-2 unfolding matrix \( X_{[n]} = X_{R_1 \times I_2 \times \cdots \times I_N} \) of \( X \) can be written as follows by the core tensors:

\[
X_{[n]} = G_{[n]}^{(2)}(G_{[n]}^{\#2})^\top
\]

(4)

where \( G_{[n]}^{(2)} \in \mathbb{R}_{+}^{I_2 \times R_n \times R_1 + 1} \) denotes the mode-2 unfolding matrix of \( G^{(n)} \) and \( G^{\#2} \in \mathbb{R}_{+}^{R_1 \times I_2 \times \cdots \times I_N} \) denotes the classical mode-2 unfolding matrix of subchain tensor \( G^{\#2} \). \( G^{\#2} \in \mathbb{R}_{+}^{R_1 \times I_2 \times \cdots \times I_N} \) is obtained by merging all core tensors except the \( n \)th core tensor as follows:

\[
G^{\#2} = G^{(n+1)}_{\times 1} \cdots \times G^{(N)}_{\times 1} \times G^{(1)}_{\times 1} \cdots \times G^{(n-1)}_{\times 1}.
\]

(5)

Based on the above analysis, the NTR problem (3) can be rewritten as the following subproblem:

\[
\min_{G_{[n]}^{(2)}} F_{\text{NTR}}^{(n)} = \frac{1}{2} \left\| X_{[n]} - G_{[n]}^{(2)}(G_{[n]}^{\#2})^\top \right\|_F^2
\]

(6)

B. Optimization of NTR

In this section, we derive the accelerated proximate gradient (APG) method to efficiently solve NTR subproblem (6). First, we derive the respective gradients of \( F_{\text{NTR}}^{(n)} \) with respect to \( G_{[n]}^{(2)} \) as follows:

\[
\frac{\partial F_{\text{NTR}}^{(n)}}{\partial G_{[n]}^{(2)}} = G_{[n]}^{(2)}(G_{[n]}^{\#2})^\top G_{[n]}^{\#2} - X_{[n]} G_{[n]}^{\#2}.
\]

(7)

It is obvious that the following propositions hold true.

**Proposition 1:** Objective function of each subproblem \( F_{\text{NTR}}^{(n)} \) is convex.

**Proposition 2:** Gradient (7) is Lipschitz continuous with the Lipschitz constant \( L_{\text{NTR}} = \| G_{[n]}^{\#2} \|^2_{\text{F}} \).

The proofs of Propositions 1 and 2 can be obtained by the proofs in [30, Appendix B]. Objective function (3) is nonconvex and cannot obtain a global optimal solution. Fortunately, the objective function of each subproblem \( F_{\text{NTR}}^{(n)} \) is convex according to Proposition 1. Hence, we can define the proximal function of \( F_{\text{NTR}}^{(n)} \) at \( G_{[n]}^{(2)} \) as follows:

\[
\phi \left( G_{[n]}^{(2)}, Y' \right) = F_{\text{NTR}}^{(n)} + \left( \frac{\partial F_{\text{NTR}}^{(n)}}{\partial G_{[n]}^{(2)}} \right) \left( G_{[n]}^{(2)} - Y' \right) + \frac{L_{\text{NTR}}}{2} \left\| G_{[n]}^{(2)} - Y' \right\|_F^2
\]

(8)

where \( t \) denotes the iteration number and \( \langle \cdot, \cdot \rangle \) denotes the inner product. Furthermore, the following sequence is constructed to select the search point \( Y' \) as follows:

\[
Y' = G_{[n]}^{(2)} + \frac{t - 1}{\alpha_t} \left( G_{[n]}^{(2)} - G_{[n]}^{(2)} \right)
\]

(9)

where \( Y' \) is denoted as the search point constructed by linearly combining two latest approximate solutions \( G_{[n]}^{(2)} \) and \( G_{[n]}^{(2)} \). The combination coefficient \( \alpha_t \) is obtained by the following equation:

\[
\alpha_{t+1} = \frac{1 + \sqrt{4\alpha_t^2 + 1}}{2}.
\]

(10)
Algorithm 1 NTR Based on the APG Algorithm

Require: Tensor $X \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_K}$, nonnegative rank $R$, the maximum number of iterations $t_{max}$.

Ensure: Nonnegative core tensors $G^{(n)}$ for $n = 1, 2, \ldots, N$.
1: Initialize $G^{(n)} \in \mathbb{R}^{K_1 \times K_2 \times \cdots \times K_{K+1}}$ for $n = 1, 2, \ldots, N$ as random tensors from the uniform distribution between 0 and 1.
2: repeat
3: for $n = 1$ to $N$ do
4: Obtain $G^{\alpha} = \left\{ G^{\alpha}_{[2]} \right\}$, $t_{NTR} = \left\| G^{\alpha}_{[2]} \right\|_2^2$, $\alpha = 1$, $Y^0 = G^{\alpha}_{[2]}$.
5: for $t = 1$ to $t_{max}$ do
6: Updating $G^{(n)(t+1)}$, $Y$ and $\alpha_{t+1}$ by Eq. (13), Eq. (9) and Eq. (10) respectively.
7: end for
8: $G^{(n)} \leftarrow$ folding $G^{(n)}_{[2]}$.
9: end for
10: until convergence.

The approximate solution of $G^{(n)(t+1)}$ can be obtained by minimizing the proximal function (8) as follows:

$$
G^{(n)(t+1)} = \arg \min_{G^{(n)}_{[2]} \geq 0} \phi(G^{(n)}_{[2]}, Y^t).
$$

By using the Lagrange multiplier method, the Karush–Kuhn–Tucker conditions of problem (11) can be expressed as follows:

$$
\begin{align*}
\frac{\partial \phi(G^{(n)}_{[2]}, Y^t)}{\partial G^{(n)}_{[2]}^t} \geq 0 & \\
\frac{\partial \phi(G^{(n)}_{[2]}, Y^t)}{\partial G^{(n)}_{[2]}} \cdot G^{(n)}_{[2]} \geq 0 & \\
\frac{\partial \phi(G^{(n)}_{[2]}, Y^t)}{\partial G^{(n)}_{[2]}} \cdot * G^{(n)}_{[2]} = 0
\end{align*}
$$

where $*$ denotes the Hadamard product. Hence, the update formula of $G^{(n)(t+1)}$ is given as follows:

$$
G^{(n)(t+1)} \leftarrow \mathcal{P}_+(Y^t - \frac{1}{t_{NTR}} \frac{\partial F^{(n)}_{\text{NTR}}}{\partial G^{(n)}_{[2]}}),
$$

where $\mathcal{P}_+(X)$ denotes the projection of the negative elements of $X$ to zero.

Based on the above analysis, the NTR method can be obtained, as shown in Algorithm 1.

C. Graph Regularization

The viewpoint of manifold learning [30], [31] takes into account that the observed high-dimensional data are actually mapped to a high-dimensional space by a low-dimensional manifold geometrical structure. Realistic high-dimensional data are often redundant, so it is difficult to observe the manifold geometrical structure in the data. Fortunately, the neighbor graph has been verified to effectively characterize manifold geometrical structures [32]. Based on this idea, the manifold geometrical information is encoded by connecting each tensor subject with its $p$-nearest neighbors in the following:

$$
W_j = \begin{cases} 
1, & \text{if } x_i \in \mathcal{N}_p(x_j), \text{ and } x_j \in \mathcal{N}_p(x_i) \\
0, & \text{otherwise}
\end{cases}
$$

where $\mathcal{N}_p(x_i)$ denotes the set of $p$ objects that are the closest to the $i$th object $x_i$.

With the weight matrix $W$, the following term [33], [34] can be used to measure the smoothness of the low-dimensional representation of the NTR model:

$$
\begin{align*}
F_G &= \frac{1}{2} \sum_{i,j=1}^{I_1} \| g_i - g_j \|_2^2 W_{ij} \\
&= \sum_{i=1}^{I_1} g_i^T D_i g_i - \sum_{i,j=1}^{I_1} g_i^T g_j W_{ij} \\
&= \text{Tr} \left( \left( G^{(N)}_{[2]} \right)^T D G^{(N)}_{[2]} \right) - \text{Tr} \left( \left( G^{(N)}_{[2]} \right)^T W G^{(N)}_{[2]} \right) \\
&= \text{Tr} \left( \left( G^{(N)}_{[2]} \right)^T H g G^{(N)}_{[2]} \right)
\end{align*}
$$

where $\text{Tr}(\cdot)$ is defined as the trace of a matrix, $D = \sum W_{ij}$. $H = D - W$.

D. Joint Objective Function of GNTR

In this section, we propose the GNTR model by incorporating manifold geometric information of tensor data in the NTR model. The GNTR model minimizes the following joint objective function:

$$
\begin{align*}
\min_{G^{(n)}_{[2]}} F^{(n)}_{\text{GNTR}} = \frac{1}{2} \left\| X_{[n]} - G^{(n)}_{[2]} \left( G^{\alpha}_{[2]} \right)^T \right\|_F^2 + \frac{\beta}{2} \text{Tr} \left( \left( G^{(N)}_{[2]} \right)^T H g G^{(N)}_{[2]} \right) \\
\text{s.t. } G^{(n)}_{[2]} \geq 0, G^{\alpha}_{[2]} \geq 0, n = 1, 2, \ldots, N
\end{align*}
$$

where $\beta \geq 0$ is a parameter that controls the intensity of the graph regularization term. By minimizing the graph regularization term, we hope that if two objects are close in the manifold space, their low-dimensional features can close to each other. We simply let the $N$th mode of $\mathcal{X}$ denotes the numbers of sample, and thus graph regularization is naturally applied to $G^{(N)}_{[2]}$, without loss of generality. Fig. 3 illustrates how the GNTR model is able to extract the parts-based components with meaningful color and line patterns from the faces images included in the Faces94 database.

E. Optimization of GNTR

In this section, we design the APG method to solve the GNTR problem. We derive the gradient of $F^{(n)}_{\text{GNTR}}$ with respect to $G^{(n)}_{[2]}$ in the case of $n = N$ as follows:

$$
\frac{\partial F^{(n)}_{\text{GNTR}}}{\partial G^{(N)}_{[2]}} = G^{(N)}_{[2]} \left( G^{\alpha}_{[2]} \right)^T G^{\alpha}_{[2]} - X_{[n]} G^{(N)}_{[2]} + \beta H g G^{(N)}_{[2]}.
$$

It is obvious that the following propositions hold true. 

Proposition 3: The objective function of each subproblem $F^{(n)}_{\text{GNTR}}$ is convex.

Proposition 4: Gradient (16) is Lipschitz continuous with the Lipschitz constant $L_{\text{GNTR}} = \left\| \left( G^{(N)}_{[2]} \right)^T G^{\alpha}_{[2]} \right\|_2 + \| \beta H g \|_2$. 

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Algorithm 2 GNTR Based on the APG Algorithm

Require: Tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, nonnegative rank $r_n$ for $n = 1, 2, \ldots, N$, the maximum number of iterations $t_{\text{max}}$, the balance parameter $\beta$.

Ensure: Core tensors $G^{(n)}$ for $n = 1, 2, \ldots, N$.

1. Initialize $G^{(n)} \in \mathbb{R}^{R_n \times I_n \times R_{n+1}}$ for $n = 1, 2, \ldots, N$ as random tensors from the uniform distribution between 0 and 1.

2. repeat
   3. for $n = 1$ to $N$ do
      4. Obtain $G^{(n)}$ by minimizing the function (17) under $G^{(n)} G^{(n)} = L_{\text{NTR}} G^{(n)} + \|\beta H_n\|_2$, $G^{(n)} = G^{(n)}$.
      5. if $n = 1$ then
         6. Updating $G^{(n)}$ by Eq. (18).
      7. else
         8. Updating $G^{(n)}$ by Eq. (13).
      9. end if
   10. end for
   11. Updating $Y'$ and $\alpha_{t-1}$ by Eq. (9), Eq. (10), respectively.
   12. $G^{(n)}$ ← folding $(G^{(n)}).$
   13. end for
   14. until convergence.

The proofs of Propositions 3 and 4 can be obtained by the proofs in [30, Appendix B]. Hence, the GNTR method based on the APG method can be developed.

The graph regularization term only affects the core tensor $G^{(N)}$ of GNTR. Therefore, in the case of $n = N$, the proximal function of $\mathcal{T}_{\text{GNTR}}$ at $G^{(N)}$ is defined as follows:

$$\phi(G^{(N)}(2), Y') = \mathcal{T}_{\text{GNTR}}(G^{(N)}(2), Y') = \frac{\partial \mathcal{T}_{\text{GNTR}}}{\partial G^{(N)}(2)} G^{(N)}(2) - Y' + L_{\text{GNTR}} \left\| G^{(N)}(2) - Y' \right\|_F^2_g = (17)$$

where the search point $Y'$ is obtained by (9), and the combination coefficient $\alpha$ is obtained by (10). Then, $G^{(N)}(2)$ is obtained by minimizing the function (17) under $G^{(N)}(2) \geq 0$ as follows:

$$G^{(N+1)}(2) \leftarrow \mathcal{T}_{\text{GNTR}}(Y' - \frac{1}{L_{\text{GNTR}}} \frac{\partial \mathcal{T}_{\text{GNTR}}}{\partial G^{(N)}(2)}).$$

In summary, the GNTR method can be obtained, as shown in Algorithm 2.

F. Convergence Analysis

As problem (3) is nonconvex, the global optimal solution cannot be obtained. For the alternating non-negative least squares method, no matter how many block variables there are, Paatero [35] proved that the method can only converge if there is a unique solution for each subproblem. In this section, we prove the APG method can guarantee the weak convergence property under some mild conditions for the NTR and GNTR problem, which is similar to the previous work [30].

Proposition 5: Suppose that $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is an $N$th-order tensor with TR ranks $R = [R_1, R_2, \ldots, R_N]$ and $R_1 = R_{N+1}$; that is, $\mathcal{X} = \text{TR}(G^{(1)}, G^{(2)}, \ldots, G^{(N)}), G^{(n)} \in \mathbb{R}^{R_n \times I_n \times R_{n+1}}$, and $n = 1, 2, \ldots, N$. Then, we have $X = (G^{(n)} G^{(n)})$. If $R_n R_{n+1} \leq I_n$ and all $G^{(n)}$ are full rank, we have $R(X) = R_n R_{n+1}$, and thus, $G^{(n)}$ is full rank.

Proof: By assuming that $G^{(n)}$ is not full rank, we simply let $G^{(n)} = \tilde{G}^{(n)T} U^{(n)}$, where $G^{(n)} \in \mathbb{R}^{I_n \times R_n \times R_{n+1}}$, $U^{(n)} \in \mathbb{R}^{R_n \times R_{n+1}}$. Then, we have $X = G^{(n)} G^{(n)} = G^{(n)} U^{(n)} G^{(n)T} = \tilde{G}^{(n)T} U^{(n)} G^{(n)T}$, where $\tilde{G}^{(n)} = G^{(n)} \tilde{U}^{(n)} \in \mathbb{R}^{I_n \times R_{n+1}}$. Hence, we have a new TR decomposition $\mathcal{X} = \text{TR}(\tilde{G}^{(1)}, \tilde{G}^{(2)}, \ldots, \tilde{G}^{(N)})$, where $\tilde{G}^{(n)} \in \mathbb{R}^{R_n \times I_n \times R_{n+1}}$, obviously, the new TR decomposition demonstrates that $R(X) = R_{n+1} + R_n - 1$, which is full rank when $R_n R_{n+1} \leq I_n$. Therefore, each subproblem $F^{(n)}$ is strictly convex. This completes the proof.

Theorem 2: Let $G^{(n)}$, $n = 1, 2, \ldots, N$ be full rank and $R_n R_{n+1} \leq I_n$; that is, $R(G^{(n)}) = R_n R_{n+1}$. Each subproblem $F^{(n)}$ can converge to a unique and optimal solution, which indicates that local convergence is guaranteed.

Proof: The Hessian matrix of $F^{(n)}$ is derived as follows:

$$\frac{\partial F^{(n)}}{\partial G^{(n)}} = E_u \otimes (G^{(n)} G^{(n)})$$

where $E_u \in \mathbb{R}^{R_n \times R_{n+1}}$ denotes an identity matrix. It is obvious that $F^{(n)}$ is strictly convex only if $(G^{(n)} G^{(n)})$ is positive definite. According to Proposition 5, we find that $G^{(n)}$ is full rank when $R_n R_{n+1} \leq I_n$. Therefore, each subproblem $F^{(n)}$ is strictly convex. This completes the proof.

The convergence of the GNTR method can be derived. We find that the Hessian matrix of each subproblem (15) is positive definite. Therefore, each subproblem $F^{(n)}$ is strictly convex, and the convergence of the GNTR method can be proven.

G. Computational Complexity Analysis

In this section, we discuss the computational complexity of our proposed NTR and GNTR methods. For simplicity, we discuss the complexity under the conditions that $I = I_n$, $R = R_n R_{n+1}, n = 1, 2, \ldots, N$ and the number of iterations $t_{\text{max}}$ at convergence. Note that for $(G^{(n)} G^{(n)})$, we only need to perform contraction operations on two identical small core tensors sequentially and then compute their matrix product. This implicit calculation requires $(N-1)R^2 + (N-2)R^3$ floating-point multiplication, which increases linearly with respect to the order of $\mathcal{X}$.

We list the computational operation counts in Table II. Compared with the cost of the NTR method, the cost of the GNTR method is only increased at optimizing the graph regularization term. As $H_X$ is usually sparse, the computational complexity of the graph regularization term is less than $O(Ip^2)$, where $p$ is the number of nearest neighbors, which is almost negligible.

IV. TASKS

In this section, we compare our proposed NTR and GNTR methods with the state-of-the-art methods. All tasks were performed on a Windows 7 machine with an i7 CPU at 3.40 GHz and 16 GB memory.
A. Databases

The following databases were used in our task.

1) ORL Database: The ORL database consists of 40 grayscale 112 × 92 face images of 40 distinct subjects. It has ten different images for each individual at different times with varying lighting, facial expressions, and facial details. We adjusted the resolution of each image to 32 × 27 and constructed a 3rd-order tensor \( T \in \mathbb{R}_{32 \times 27}^{300 \times 1} \).

2) FEI PART 1 Database: The FEI PART 1 database is a subset of the FEI database and consists of 700 color images of size 480 × 640 × 3 collected from 50 individuals. There are 14 different images of each individual with different views and facial expressions. Each image is downsampled to 48 × 64 pixels and we can construct a 4th-order tensor \( T \in \mathbb{R}_{48 \times 64}^{640 \times 3} \).

3) GT Database: The Georgia Tech database contains 750 color images of 50 people, and each image has a resolution of 640 × 480 pixels. The backgrounds of the images are messy. The faces have different orientations, and the images contain also have different facial expressions, lighting conditions, and proportions. After downsampling each image to 40 × 30 × 3, we obtain a 4th-order tensor \( T \in \mathbb{R}_{40 \times 30 \times 3}^{480 \times 3} \).

4) COIL-100 PART 1 Database: The Columbia Object Image Library (COIL-100) database contains 7200 color images of 100 objects. There are 72 images of each object taken from different poses. The size of each image is 128 × 128 × 3. We only considered the first 20 categories and resized all the images into 32 × 32 × 3. Therefore, we obtain the COIL-100 PART 1 database, which is a 4th-order tensor \( T \in \mathbb{R}_{32 \times 32 \times 3}^{1000 \times 3} \).

5) Faces94 PART 1 Database: The Faces94 database consists of 3060 color images of 153 individuals in permanent positions with respect to the camera for a total of 20 facial expressions. We used images of the first 72 individuals and downsampled them to 50 × 45 × 3 pixels. Finally, we obtain the Faces94 PART 1 database as a 4th-order tensor \( T \in \mathbb{R}_{50 \times 45 \times 3}^{450 \times 3} \).

6) STL-10 Database: The STL-10 database consists of ten objects with 5000 training and 8000 test images. Each object has 1300 images of size 96 × 96 × 3. After downsampling them to 32 × 32 × 3 pixels, we obtain a 4th-order tensor \( T \in \mathbb{R}_{32 \times 32 \times 3}^{1300 \times 3} \).

B. Comparison Methods

We compare our methods with the following state-of-the-art methods in the tasks.

1) Principal Component Analysis (PCA) [36]: PCA is a famous unsupervised dimensionality reduction method.

2) Graph-Laplacian PCA (gLPCA) [37]: gLPCA is a PCA method that considers the geometric structure information of the data.

3) NMF [1]: NMF aims to learn the non-negative parts-based basis of data objects.

4) Graph-Regularized Non-Negative Matrix Factorization (GNMF) [33]: GNMF is an NMF method that considers the geometric structure information of the data.

5) NTF [15]: Non-negative tensor factorization (NTF) is a non-negative representation method for tensor data based on the CP structure.

6) Laplacian-Regularized NTF (LRNTF) [38]: LRNTF is the NTF method that considers the geometric structure information of the data.

7) NTD [14]: NTD is a non-negative representation method based on the Tucker structure.

8) Graph-Regularized NTD (GNTD) [39]: GNTD is the NTD method that considers the manifold geometric structure of data.

9) NTT [20]: NTT decomposition represents tensor data as a series of non-negative low-rank core tensors and the APG method is adopted to optimize the NTT model.

10) Graph-Regularized NTT (GNTT): GNTT decomposition is an NTT model that is combined with a graph-regularized term. The APG method is adopted to optimize the GNTT model.
C. Evaluation Metrics

To compare the sparseness of the basis extracted by each method, the sparseness level metric [40] is adopted in this task, which is defined as follows:

\[
\text{Sparseness}(\mathbf{S}) = \frac{\sqrt{n} - \|\text{vec}(\mathbf{S})\|_1/\|\text{vec}(\mathbf{S})\|_2}{\sqrt{n} - 1}
\]

(20)

where \(\mathbf{S}\) and \(\text{vec}(\mathbf{S})\) denote the basis extracted by the method and its vectorization, respectively. \(n\) denotes the number of elements in \(\mathbf{H}\). \(\| \cdot \|_1\) and \(\| \cdot \|_2\) are defined as the \(L_1\) and \(L_2\) norms, respectively.

To quantitatively evaluate the effectiveness of each method, we adopted three metrics: 1) accuracy (ACC); 2) normalized mutual information (NMI); and 3) purity (PUR). The definition of ACC is as follows:

\[
\text{ACC}(y_i, \hat{y}_i) = \frac{1}{n} \sum_{i=1}^{n} \delta(y_i, \text{map}(\hat{y}_i))
\]

(21)

where \(n\) is the total number of objects. \(y_i\) and \(\hat{y}_i\) represent the cluster label of the object and the true label of the object, respectively. \(\text{map}(\cdot)\) denotes a displacement mapping function, which is responsible for mapping each cluster label \(y_i\) to the equivalent label from the data corpus. If the object label \(y_i\) and the real label \(\hat{y}_i\) are equal, then \((y_i, \text{map}(\hat{y}_i)) = 1\), if not, then \((y_i, \text{map}(\hat{y}_i)) = 0\).

By employing the information theory, the agreement between two cluster partitions can be measured with mutual information (MI). The MI between the set of cluster labels \(C\) and the set of true labels \(C'\) is defined as follows:

\[
\text{MI}(C, C') = \sum_{c_i \in C} \sum_{c_j' \in C'} p(c_i, c_j') \cdot \log_2 \frac{p(c_i, c_j')}{p(c_i) \cdot p(c_j')}
\]

(22)

where \(p(c_i)\) and \(p(c_j')\) denote the probabilities that the objects belong to categories \(c_i\) and \(c_j'\), respectively, by randomly selecting an object from the databases. \(p(c_i, c_j')\) is defined as the probability that an object belongs to categories \(c_i\) and \(c_j'\) at the same time by randomly selecting an object from the databases. To force the score to have an upper bound, we used NMI as one of the evaluation measures, and the definition of NMI is denoted as follows:

\[
\text{NMI}(C, C') = \frac{\text{MI}(C, C')}{\max(H(C), H(C'))}
\]

(23)

where \(H(C)\) and \(H(C')\) are defined as the entropy of the true label set \(C\) and the entropy of the cluster label set \(C'\). It is quite straightforward to know that the score ranges of NMI(\(C, C'\)) is from 0 to 1, where NMI(\(C, C'\)) = 1 if the two label sets are the same, and NMI(\(C, C'\)) = 0 otherwise.

To measure the extent to which each cluster contains data points from primarily one class, the PUR score of clustering is observed by the weighted sum of the individual cluster PUR values, and the definition of PUR is as follows:

\[
\text{PUR}(\Omega, C) = \frac{1}{n} \sum_{k} \max\{|w_k \cap c_j|\}
\]

where \(C = \{c_1, c_2, \ldots, c_k\}\) and \(C' = \{c_1', c_2', \ldots, c_j'\}\) denote the set of cluster and true category.

D. Basis Visualization

The non-negativity of data representation brings about two key effects [16]: 1) the data representation is purely additive and 2) it can provide the more interpretable and meaningful components. The data representation is usually sparse. Due to these effects, the non-negative matrix/tensor decomposition methods have the ability to extract the parts-based components from non-negative tensor objects.

In order to observe the components of the data extracted by different methods, we visualize the basis extracted from the Faces94 PART 1 database by each method. The number of basis is set to the number of categories 72 of the Faces94 PART 1 database. For the matrix-based methods, we set the rank as \(R = 72\). For the tensor-based methods except for NTR and GNTR, we set the non-negative rank of their feature matrix/tensor to \(k = 72\). In particular, since the feature core tensor of NTR and GNTR has two non-negative ranks \(R_1\) and \(R_4\), we set \(R_1 \times R_4 = 72\) to extract 72 basis from the database. For the tensor-based methods except for NCP and LRNTF, we set the rest of the non-negative rank uniformly to 10, so that all tensor-based methods can have a similar number of parameters after decomposition. For the graph-based methods, we empirically set the graph regularization parameter \(\beta = 0.1\) and the number of nearest neighbors as \(k = 5\). After obtaining the 72 basis extracted by all methods, we stitched these basis images together and visually displayed them in Fig. 4. And we also report the sparseness of the basis extracted by different methods in Fig. 5. The experimental results are summarized as follows.

1) Both the NTR and GNTR models have a flexible and balance structure with a non-negative constraint, and they feature core tensor \(G^{(4)}\) can maintain direct connections and interactions with the first core tensor \(G^{(1)}\). Therefore, NTR and GNTR can extract parts-based components with meaningful color and line patterns from the Faces94 database.

2) NTF and LRNTF extract the sparse basis of the tensor objects by considering the high-dimensional structures in the image space, which is consistent with the results [15], [38]. However, the lines of the components extracted by the NTF model are too abstract to identify the outline of the tensor object. The reason for this possibility is that NTF and LRNTF assume that the rank of the different modes of the tensor data is the same [25].

3) NTT and GNNTT extract the basis with color pattern from the Faces94 PART 1 database. The reason is that the relatively unbalanced structure of NTT restricts the interaction of the feature core tensor with the first core tensor that stores row pixel information.

4) For NTR and GNTR, the extracted basis with blue patterns indicates the component of light noise in the dataset (blue light on some human faces in Fig. 4). There is also a part of the basis of the green pattern, corresponding to the common features in the samples (shooting background). It can be well recognized that the remaining basis depicts the details of the human faces.
It can be shown that the results obtained by our methods are interpretable.

To investigate the effect of different non-negative ranks in the feature core tensor $G(N)$ of NTR on the extracted components, we visualized the basis representation extracted from the COIL-100 database by NTR under different combinations of two non-negative ranks of the feature core tensor in Fig. 6. The results have brought about the discovery of some new phenomenons.

1) As $R_3$ increases, the color patterns of the components extracted by the NTR model increase.
2) As $R_4$ increases, the line patterns of the components extracted by the NTR model increase.

These interesting phenomenons have never been observed in previous related research. The main reason for the phenomena is that the non-negativity structure of NTR is more flexible and the feature core tensor of the NTR can remain connected and interact with the adjacent core tensor. Therefore, we can obtain a basis with different color and line diversities by controlling the rank combination of the feature core tensor, which provides a more interpretable and meaningful representation.

In practical applications, to enhance the effectiveness of data representation from tensor objects with rich colors, we can highlight the diversity of color on a basis by choosing the special non-negative rank combination. To better depict the details of different objects from the tensor objects with rich lines, we can highlight the diversity of the line in the basis by choosing the special non-negative rank combination.

E. Clustering and Classification Tasks

To verify the performance of our proposed methods in data representation, we compared them with the state-of-the-art methods in clustering and classification tasks.

1) Clustering Task: For simplicity, each method extracts a feature vector of length $c$ from the tensor data, where $c$ is the number of categories in the database. For the matrix-based methods, such as PCA, gLPCA, NMF, and GNMF, we set the rank as $R = c$ to extract $c$ features of each object. Similarly, for the CP-based methods, such as NCP and LRNTF, we set the non-negative rank as $R = c$. For Tucker-based methods, such as NTD and GNTD, we empirically set the non-negative rank as $R_1 = R_2 = 10$, $R_3 = 3$, and $R_4 = c$ for the task on the 4th-order database and set $R_1 = R_2 = 10$ and $R_3 = k$ for the task on the 3rd-order database. For the TT-based methods, such as NTT and GNTT, we set the last non-negative rank to $c$ and empirically set the range of other non-negative ranks to $\{2, 3, 4\}$, and report the experimental results using the rank combination with the best performance. It is worth mentioning that for the TR-based methods, such as TR, NTR, and GNTR, we set two multiway ranks of the last core tensor as a pair of integer factors of $c$, and the other multiway ranks are empirically set to 2. In most graph-based methods, in addition to gLPCA, the regularization parameter $\beta$ is set as 0.1. For the gLPCA, the $\beta$ is set as 0.01 empirically. For all the graph-based methods, the number of nearest neighbors is set as 5. To mitigate the local convergence issue, we repeat the process 200 times with random initialization for each $K$-means run. For each database, we repeat the above experimental process of the task ten times and report the average performance in Tables III–V. From the experimental results, we can draw the following conclusions.
Fig. 6. Visualization of the objects of the COIL-100 database and the visualization of the basis extracted of NTR under different combinations of non-negative rank of the feature core tensor $G^{(4)} \in \mathbb{R}^{4 \times 2 \times 80}$. (a) Visualization of the objects of the COIL-100 database. The different size of the feature core tensor $G^{(4)}$. (b) $G^{(4)} \in \mathbb{R}^{2 \times 720 \times 50}$. (c) $G^{(4)} \in \mathbb{R}^{10 \times 720 \times 10}$. (d) $G^{(4)} \in \mathbb{R}^{5 \times 720 \times 20}$. (e) $G^{(4)} \in \mathbb{R}^{1 \times 720 \times 50}$. (f) $G^{(4)} \in \mathbb{R}^{30 \times 720 \times 5}$. 

TABLE III
ACC (MEAN % ± STD %) OF DIFFERENT METHODS ON SIX DATABASES IN THE CLUSTERING TASK. THE BEST RESULTS ARE MARKED IN BOLD FONT AND THE SECOND BEST RESULTS ARE MARKED BY UNDERLINED

| Method   | ORL | FEI PART 1 | GT | COIL-100 PART 1 | Faces94 PART 1 | STL-10 |
|----------|-----|------------|----|----------------|----------------|--------|
| Original | 83.16 ± 1.52 | 79.83 ± 1.53 | 82.62 ± 0.81 | 80.53 ± 0.32 | 81.93 ± 1.18 | 73.43 ± 0.14 |
| PCA      | 78.55 ± 1.78 | 75.04 ± 1.03 | 79.51 ± 1.57 | 79.41 ± 0.96 | 85.56 ± 0.24 | 75.64 ± 0.17 |
| gPCA     | 80.17 ± 1.78 | 78.03 ± 1.43 | 80.85 ± 1.36 | 78.78 ± 0.70 | 88.10 ± 1.63 | 77.49 ± 0.20 |
| NMF      | 82.83 ± 0.70 | 80.34 ± 0.34 | 83.26 ± 1.34 | 81.17 ± 0.77 | 91.90 ± 1.63 | 79.50 ± 0.20 |
| GSNMF    | 82.52 ± 1.15 | 80.54 ± 0.44 | 83.26 ± 0.96 | 81.17 ± 0.77 | 91.90 ± 1.63 | 79.50 ± 0.20 |
| NTD      | 81.20 ± 1.09 | 79.73 ± 1.82 | 81.69 ± 0.93 | 80.12 ± 0.68 | 91.90 ± 1.63 | 79.50 ± 0.20 |
| LRNTF    | 87.21 ± 0.88 | 84.76 ± 0.99 | 87.49 ± 0.98 | 84.76 ± 0.99 | 91.90 ± 1.63 | 79.50 ± 0.20 |
| NTR      | 86.66 ± 0.85 | 84.62 ± 0.32 | 86.47 ± 0.57 | 85.30 ± 0.63 | 92.16 ± 0.88 | 79.50 ± 0.20 |
| GNTR     | 87.76 ± 0.85 | 84.87 ± 0.72 | 86.69 ± 0.52 | 85.16 ± 0.88 | 92.16 ± 0.88 | 79.50 ± 0.20 |

TABLE IV
NMI (MEAN % ± STD %) OF DIFFERENT METHODS ON SIX DATABASES IN THE CLUSTERING TASK. THE BEST RESULTS ARE MARKED IN BOLD FONT AND THE SECOND BEST RESULTS ARE MARKED BY UNDERLINED

| Method   | ORL | FEI PART 1 | GT | COIL-100 PART 1 | Faces94 PART 1 | STL-10 |
|----------|-----|------------|----|----------------|----------------|--------|
| Original | 83.16 ± 1.52 | 79.83 ± 1.53 | 82.62 ± 0.81 | 80.53 ± 0.32 | 81.93 ± 1.18 | 73.43 ± 0.14 |
| PCA      | 78.55 ± 1.78 | 75.04 ± 1.03 | 79.51 ± 1.57 | 79.41 ± 0.96 | 85.56 ± 0.24 | 75.64 ± 0.17 |
| gPCA     | 80.17 ± 1.78 | 78.03 ± 1.43 | 80.85 ± 1.36 | 78.78 ± 0.70 | 88.10 ± 1.63 | 77.49 ± 0.20 |
| NMF      | 82.83 ± 0.70 | 80.34 ± 0.34 | 83.26 ± 1.34 | 81.17 ± 0.77 | 91.90 ± 1.63 | 79.50 ± 0.20 |
| GSNMF    | 82.52 ± 1.15 | 80.54 ± 0.44 | 83.26 ± 0.96 | 81.17 ± 0.77 | 91.90 ± 1.63 | 79.50 ± 0.20 |
| NTD      | 81.20 ± 1.09 | 79.73 ± 1.82 | 81.69 ± 0.93 | 80.12 ± 0.68 | 91.90 ± 1.63 | 79.50 ± 0.20 |
| LRNTF    | 87.21 ± 0.88 | 84.76 ± 0.99 | 87.49 ± 0.98 | 84.76 ± 0.99 | 91.90 ± 1.63 | 79.50 ± 0.20 |
| NTR      | 86.66 ± 0.85 | 84.62 ± 0.32 | 86.47 ± 0.57 | 85.30 ± 0.63 | 92.16 ± 0.88 | 79.50 ± 0.20 |
| GNTR     | 87.76 ± 0.85 | 84.87 ± 0.72 | 86.69 ± 0.52 | 85.16 ± 0.88 | 92.16 ± 0.88 | 79.50 ± 0.20 |

1) The proposed GNTR method is superior to all other methods in most cases.
2) Compared with TR, NTR achieves better performance in most cases, which demonstrates that non-negative constraints can make the data representation more discriminative.
3) Graph-based methods, such as GNMF, LRNTF, and GNTD, achieve better performance than the corresponding methods without graph regularization, such as NMF, NTF, and NTD. This phenomenon can also be observed in [38], [39], and [34] and suggests the importance of learning the geometrical structure information of tensor data.

2) Classification Task: We randomly select 20% and 40% of objects for each class as training data, and the other data are used as the testing data. For the number of features extracted from the database, all methods are uniformly set to extract the number of categories $c$ form on each database, except for STL-10, which is set to extract $2k$ features of each sample.

For the setting of the other ranks of the tensor-based method, we uniformly follow Section IV-E1. For all the graph-based methods, we set the number of nearest neighbors as 5 and perform grid search based on the development set by using $\beta \in \{0.001, 0.01, 0.1\}$. The results of some methods have large standard deviations (STDs) in this setting, which indicates overfitting. In this case, we improved the results by adjusting the parameters.
the feature number. We perform the classification task (see [3]) for all methods. Take our proposed NTR and GNTR methods as an example, we first decomposed the training data and then obtained the feature core tensor \( G^{(N)} \) and the basis tensor \( G^{(N)} = G^{(1)} \times G^{(2)} \times \ldots \times G^{(N-1)} \). Second, we trained the \( k \)-nearest neighbor (\( k \)-NN) classifier on \( G^{(N)} \). Third, we projected the test data onto \( G^{(N)} \) to obtained the features of the test data and then used the \( k \)-NN classifier to recognize the features of the test data. Finally, we repeated the above task five times and reported the average classification accuracy.

From Tables VI–XI, it can be observed that the proposed GNTR performs the best in most cases, which further proves the advantages of the GNTR method in the multivariate representation of tensor data.

It is worth noting that NTT achieves good performance in the task on the COIL-100 PART 1 database and conversely achieves poor performance on the FEI PART 1 database. And the performances of NTR and GNTR are better than NTT and GNTT in most cases, respectively. One possible reason is that our methods have a greater degree of freedom in their rank and are able to treat all modes of tensor equally. Therefore, NTR and GNTR can comprehensively take into account the similarities and differences of the line and color patterns of the tensor object to achieve better performance in the classification tasks.
TABLE VIII

| Method | ORL | PRD PART I | GT | COL-CCO PART | FaceRec PART I | STL-10 | ORL | PRD PART I | GT | COL-CCO PART | FaceRec PART I | STL-10 |
|--------|-----|------------|----|--------------|---------------|--------|-----|------------|----|--------------|---------------|--------|
| Original | 70.37 ± 0.60 | 74.67 ± 0.11 | 54.71 ± 1.30 | 84.83 ± 0.58 | 57.60 ± 0.49 | 29.48 ± 0.28 | 74.67 ± 0.11 | 54.71 ± 1.30 | 84.83 ± 0.58 | 57.60 ± 0.49 | 29.48 ± 0.28 |
| PCA | 72.23 ± 0.18 | 76.13 ± 0.01 | 57.91 ± 0.09 | 85.36 ± 1.07 | 59.51 ± 0.93 | 33.34 ± 0.37 | 76.13 ± 0.01 | 57.91 ± 0.09 | 85.36 ± 1.07 | 59.51 ± 0.93 | 33.34 ± 0.37 |
| bLPCA | 72.42 ± 0.36 | 76.35 ± 1.74 | 55.82 ± 1.27 | 84.64 ± 0.48 | 57.37 ± 0.39 | 33.38 ± 0.31 | 76.35 ± 1.74 | 55.82 ± 1.27 | 84.64 ± 0.48 | 57.37 ± 0.39 | 33.38 ± 0.31 |
| NMF | 72.67 ± 0.58 | 75.38 ± 1.95 | 56.18 ± 1.29 | 84.17 ± 0.71 | 57.98 ± 0.35 | 33.41 ± 0.37 | 75.38 ± 1.95 | 56.18 ± 1.29 | 84.17 ± 0.71 | 57.98 ± 0.35 | 33.41 ± 0.37 |
| GNMF | 71.05 ± 1.71 | 55.69 ± 2.00 | 55.69 ± 1.32 | 83.35 ± 1.39 | 57.94 ± 1.47 | 33.42 ± 0.37 | 55.69 ± 2.00 | 55.69 ± 1.32 | 83.35 ± 1.39 | 57.94 ± 1.47 | 33.42 ± 0.37 |
| NTP | 89.22 ± 0.68 | 85.86 ± 1.53 | 52.27 ± 1.47 | 87.59 ± 0.76 | 70.66 ± 0.25 | 23.74 ± 1.73 | 85.86 ± 1.53 | 52.27 ± 1.47 | 87.59 ± 0.76 | 70.66 ± 0.25 | 23.74 ± 1.73 |
| LSNMF | 70.32 ± 0.73 | 76.60 ± 1.59 | 53.24 ± 1.41 | 84.37 ± 0.73 | 57.29 ± 0.38 | 20.64 ± 1.41 | 76.60 ± 1.59 | 53.24 ± 1.41 | 84.37 ± 0.73 | 57.29 ± 0.38 | 20.64 ± 1.41 |
| kNN | 87.99 ± 0.67 | 72.19 ± 2.00 | 55.69 ± 1.32 | 83.35 ± 1.39 | 57.94 ± 1.47 | 33.42 ± 0.37 | 72.19 ± 2.00 | 55.69 ± 1.32 | 83.35 ± 1.39 | 57.94 ± 1.47 | 33.42 ± 0.37 |
| GNTP | 70.42 ± 0.58 | 73.45 ± 2.77 | 54.62 ± 2.14 | 85.37 ± 0.54 | 57.93 ± 0.60 | 33.34 ± 0.74 | 73.45 ± 2.77 | 54.62 ± 2.14 | 85.37 ± 0.54 | 57.93 ± 0.60 | 33.34 ± 0.74 |
| NTR | 64.00 ± 1.68 | 72.85 ± 2.19 | 53.24 ± 2.66 | 73.31 ± 2.23 | 45.19 ± 0.87 | 24.19 ± 1.07 | 72.85 ± 2.19 | 53.24 ± 2.66 | 73.31 ± 2.23 | 45.19 ± 0.87 | 24.19 ± 1.07 |
| GNT | 58.43 ± 2.04 | 68.42 ± 2.69 | 60.88 ± 3.56 | 65.87 ± 1.07 | 59.52 ± 1.39 | 33.66 ± 0.88 | 68.42 ± 2.69 | 60.88 ± 3.56 | 65.87 ± 1.07 | 59.52 ± 1.39 | 33.66 ± 0.88 |

F. Computational Cost

Fig. 7 shows the average running times of all the methods when decomposing the training data in the classification task. From Fig. 7, we can observe that the matrix-based method is usually faster than the tensor-based method because the extra unfolding and folding operations of the tensor-based method are time consuming. Nonetheless, our proposed NTR and GNTR usually perform faster than the other non-negative tensor-based methods. The reason is that the NTR and GNTR model can alleviate the curse of dimensionality and only require significantly smaller ranks than the other models in practice. Therefore, our proposed NTR and GNTR methods are more efficient in practice.
H. Convergence Study

In this section, we show the convergence curves of the proposed NTR and GNTR methods on the six databases in Fig. 9. The experimental results demonstrate that the proposed methods converge very quickly and usually converge within 150 iterations.

I. Discussion

Our proposed methods have achieved satisfying results in the task, but we need to discuss the following issues for future work. The major limitation of the proposed methods is that we have to choose the non-negative ranks manually. In this article, we set these parameters empirically. How to obtain the optimal ranks is still an open and challenging problem.

V. Conclusion

In this article, we proposed the NTR model that represented the non-negative high-order tensor as circular contractions over a sequence of the 3rd-order core tensors. It extracts the meaningful parts-based components with color and line patterns from the objects and thus provides the interpretable result for physical signals. We also combine graph regularization with NTR to develop GNTR, which perfectly inherits the advantages of NTR and captures the manifold geometrical structure of tensor data. An efficient algorithm based on the APG is developed to optimize our proposed models, and the convergence of our methods is discussed and validated. The experimental results demonstrate the effectiveness of our proposed methods.

REFERENCES

[1] D. D. Lee and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization,” Nature, vol. 401, no. 6755, pp. 788–791, 1999.
[2] N. Gillis, “The why and how of nonnegative matrix factorization,” Jan. 2014, arXiv:1401.5226.
[3] G. Zhou, A. Cichocki, Q. Zhao, and S. Xie, “Efficient nonnegative Tucker decompositions: Algorithms and uniqueness,” IEEE Trans. Image Process., vol. 24, pp. 4990–5003, 2015.
[4] Z. Zhang and K. Zhao, “Low-rank matrix approximation with manifold regularization,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 7, pp. 1717–1729, Jul. 2013.
[5] D. Wang, X. Gao, and X. Wang, “Semi-supervised nonnegative matrix factorization via constraint propagation,” IEEE Trans. Cybern., vol. 46, no. 1, pp. 233–244, Jan. 2016.
[6] N. Liang, Z. Yang, Z. Li, S. Xie, and C.-Y. Su, “Semi-supervised multi-view clustering with graph-regularized partially shared non-negative matrix factorization,” Knowl. Based Syst., vol. 190, Feb. 2020, Art. no. 105185.
[7] Y. Su, X. Bai, P. Jian, P. Jing, and J. Zhang, “Low-rank approximation-based tensor decomposition model for subspace clustering,” Electron. Lett., vol. 55, no. 7, pp. 406–408, 2019.
[8] Y. Su, D. Hong, Y. Li, and P. Jing, “Low-rank regularized deep collaborative matrix factorization for micro-video multi-label classification,” IEEE Signal Process. Lett., vol. 27, pp. 740–744, Mar. 2020.
[9] J. Zhang, X. Li, P. Jing, J. Liu, and Y. Su, “Low-rank regularized heterogeneous tensor decomposition for subspace clustering,” IEEE Signal Process. Lett., vol. 25, no. 3, pp. 333–337, Mar. 2018.
[10] Y. Hu, J.-X. Liu, Y.-L. Gao, and J. Shang, “DSTPCA: Double-sparse constrained tensor principal component analysis method for feature selection,” IEEE/ACM Trans. Comput. Biol. Bioinf., vol. 18, no. 4, pp. 1481–1491, Jul./Aug. 2021.
[11] C.-M. Feng, Y. Yu, J.-X. Liu, Y.-L. Gao, and C.-H. Zheng, “Supervised discriminative sparse PCA for co-characteristic gene selection and tumor classification on multiview biological data,” IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 10, pp. 2926–2937, Oct. 2019.
[12] P. Jing, Y. Su, X. Jin, and C. Zhang, “High-order temporal correlation model learning for time-series prediction,” IEEE Trans. Cybern., vol. 49, no. 6, pp. 2385–2397, Jun. 2019.
[13] A. Cichocki, R. Zdunek, A. H. Phan, and S.-I. Amari, Non-Negative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation. Hoboken, NJ, USA: Wiley, 2009.
[14] Y.-D. Kim and S. Choi, “Nonnegative Tucker decomposition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Minneapolis, MN, USA, 2007, pp. 1–8.
[15] T. Hazan, S. Polak, and A. Shashua, “Sparse image coding using a 3D non-negative tensor factorization,” in Proc. IEEE Int. Conf. Comput. Vis., vol. 1. Beijing, China, 2005, pp. 50–57.
[16] G. Zhou, A. Cichocki, Q. Zhao, and S. Xie, “Nonnegative matrix and tensor factorizations: An algorithmic perspective,” IEEE Signal Process. Mag., vol. 31, no. 5, pp. 54–65, May 2014.
[17] G. Zhou, A. Cichocki, and S. Xie, “Fast nonnegative matrix/tensor factorization based on low-rank approximation,” IEEE Trans. Signal Process., vol. 60, no. 6, pp. 2928–2940, Jun. 2012.
[18] K. Fonal and R. Zdunek, “Fast recursive nonnegative standard and hierarchical Tucker decomposition,” IEEE Signal Process. Lett., vol. 26, no. 9, pp. 1265–1269, Sep. 2019.
[19] A. Cichocki, N. Lee, I. Oseledets, A.-H. Phan, Q. Zhao, and P. D. Mandic, “Tensor networks for dimensionality reduction and large-scale optimization: Part I low-rank tensor decompositions,” Found. Trends Mach. Learn., vol. 9, nos. 4–5, pp. 249–429, 2016.
[20] N. Lee, A.-H. Phan, F. Cong, and A. Cichocki, “Nonnegative tensor train decompositions for multi-domain feature extraction and clustering,” in Proc. Int. Conf. Neural Inf. Process., 2016, pp. 87–95.
[21] Q. Zhao, M. Sugiyama, L. Yuan, and A. Cichocki, “Learning efficient tensor representations with ring-structured networks,” in Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Brighton, U.K., 2019, pp. 8608–8612.
[22] W. Wang, V. Aggarwal, and S. Aeron, “Efficient low rank tensor ring completion,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Venice, Italy, 2017, pp. 5697–5705.
[23] Q. Zhao, G. Zhou, S. Xie, L. Zhang, and A. Cichocki, “Tensor ring decomposition,” 2016, arXiv:1606.05355.
[24] A. Cichocki et al., “Tensor networks for dimensionality reduction and large-scale optimizations. Part 2 applications and future perspectives,” 2017, arXiv:1708.09165.
[25] W. He, Y. Chen, N. Yokoya, C. Li, and Q. Zhao, “Hyperspectral super-resolution via coupled tensor ring factorization,” Pattern Recognit., vol. 122, Feb. 2022, Art. no. 108280.
[26] L. Yuan, C. Li, D. Mandic, J. Cao, and Q. Zhao, “Tensor ring decomposition with rank minimization on latent space: An efficient approach for tensor completion,” in Proc. AAAI Conf. Artif. Intell., vol. 33, 2019, pp. 9151–9158.
[27] Y. Chen, W. He, N. Yokoya, T.-Z. Huang, and X.-L. Zhao, “Nonlocal tensor-ring decomposition for hyperspectral image denoising,” IEEE Trans. Geosci. Remote Sens., vol. 58, no. 2, pp. 1348–1362, Feb. 2020.
[28] M. Hou, J. Tang, J. Zhang, W. Kong, and Q. Zhao, “Deep multimodal multilinear fusion with high-order polynomial pooling,” in Advances in Neural Information Processing Systems. Red Hook, NY, USA: Curran, 2019, pp. 12136–12145.
[29] Y. Zhang, Y. Sun, B. Eriksson, W. Wang, and V. Aggarwal, “Wide compression: Tensor ring nets,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Salt Lake City, UT, USA, 2018, pp. 9329–9338.
[30] Y. Qiu, G. Zhou, Y. Wang, Y. Zhang, and S. Xie, “A generalized graph regularized non-negative Tucker decomposition framework for tensor data representation,” IEEE Trans. Cybern., vol. 52, no. 1, pp. 594–607, Jan. 2022.
[31] X. Li, G. Cui, and Y. Dong, “Graph regularized non-negative low-rank matrix factorization for image clustering,” IEEE Trans. Cybern., vol. 47, no. 11, pp. 3840–3853, Nov. 2017.
[32] M. Belkin and P. Niyogi, “Laplacian Eigenmaps and spectral techniques for embedding and clustering,” in Advances in Neural Information Processing Systems. Cambridge, MA, USA: MIT Press, 2002, pp. 585–591.
[33] D. Cal, X. He, J. Han, and T. S. Huang, “Graph regularized nonnegative matrix factorization for data representation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 8, pp. 1548–1560, Aug. 2011.
Guoxu Zhou received the Ph.D. degree in intelligent signal and information processing from the South China University of Technology, Guangzhou, China, in 2010.

He was a Research Scientist with the RIKEN Brain Science Institute, Tokyo, Japan. He is currently a Full Professor with the School of Automation, Guangdong University of Technology, Guangzhou. His current research interests include statistical signal processing, tensor analysis, intelligent information processing, and machine learning.

Ning Zheng received the Ph.D. degree in information from the Graduate University for Advanced Studies, SOKENDAI, Hayama, Japan, in 2017.

He was a Project Researcher with the National Institute of Informatics and RIKEN Center for Advanced Intelligence Project, Tokyo, Japan. He is currently working with the Institute of Statistical Mathematics, Tokyo, Japan. His current research interests include numerical linear algebra, numerical optimizations, image processing, and tensor-based data analysis.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.