Uniform tensor clustering by jointly exploring sample affinities of various orders

Hongmin Cai[1*], *Senior Member, IEEE*, Fei Qi[1,2], Junyu Li[1], Yu Hu[1], Yue Zhang[3], and Bin Hu[4*], *Senior Member, IEEE*

Abstract—Conventional clustering methods based on pairwise affinity usually suffer from the concentration effect while processing huge dimensional features yet low sample sizes data, resulting in inaccuracy to encode the sample proximity and suboptimal performance in clustering. To address this issue, we propose a unified tensor clustering method (UTC) that characterizes sample proximity using multiple samples' affinity, thereby supplementing rich spatial sample distributions to boost clustering. Specifically, we find that the triadic tensor affinity can be constructed via the Khari-Rao product of two affinity matrices. Furthermore, our early work shows that the fourth-order tensor affinity is defined by the Kronecker product. Therefore, we utilize arithmetical products, Khatri-Rao and Kronecker products, to mathematically integrate different orders of affinity into a unified tensor clustering framework. Thus, the UTC jointly learns a joint low-dimensional embedding to combine various orders. Finally, a numerical scheme is designed to solve the problem. Experiments on synthetic datasets and real-world datasets demonstrate that 1) the usage of high-order tensor affinity could provide a supplementary characterization of sample proximity to the popular affinity matrix; 2) the proposed method of UTC is affirmed to enhance clustering by exploiting different order affinities when processing high-dimensional data.

Index Terms—High-order affinity, Clustering, Fusing affinity, Tensor, Spectral graph.

1 Introduction

Clustering aims to partition unlabeled data into distinct groups [1]. Traditional methods, such as K-means [2] and spectral clustering [3], [4], have been well studied, and many variants have been designed to cater to various needs. However, with advances in observatory equipment, an object of interest could be associated with high or even huge dimensional features. For example, in bioinformatics, most single cells isolated from tissue are damaged during sequencing. Thus, only a limited number of cells can be successfully sequenced, resulting in a sample matrix with a few samples and millions of gene reads [5], [6]. Similar scenarios are rising across various fields, such as computer vision [7], [8] and natural language processing. Accurate clustering high-dimension m yet low-sample-size n (HDLSS) data remains challenging when $n \gg m$ [9], [10], [11].

The critical challenge that hinders the clustering of HDLSS data is called the concentration effect [12], [13], which refers to the situation that the pairwise Euclidean distance among the samples collapses to a constant, thus resulting in the sample-wise affinity tending to be indiscriminating in high feature dimensions [14], [15], [16]. Such an effect exists as an insurmountable roadblock that hinders most clustering methods, which rely on the pairwise affinity of samples, from achieving precise clustering. Early works attempted to learn adaptive sample affinity to develop metrics driven by data. For example, CAN [17] proposed a model that dynamically learns the affinity matrix by assigning the adaptive neighbors for each data point based on local distances. However, the measurement of local distance remains a characterization of sample-wise affinity and thus still suffers from the concentration effects and performs unsatisfactorily.

To overcome such drawbacks, many researchers have proposed to incorporate the spatial distribution of multiple samples to enhance clustering. Such high-order characterization on the local structure of multiple points can effectively overcome the shortcomings of pairwise similarity and enhance the clustering effect [18] [19]. For example, [20] generalizes the spectral methods operating on a pairwise similarity graph to a hypergraph that can represent high-order affinities and develops an algorithm to obtain the hypergraph embedding for clustering. [21] proposes to maintain a weight graph to approximate the hypergraph generated by high-order association and then use a spectral method to accomplish graph partitioning. [22] considers the hypergraph clustering problem as a multiplayer noncooperative game from a game-theoretic approach. Like the CAN method, [23] first proposes a way to learn the dynamic hypergraph for clustering. However, the above method must eventually convert the hypergraph into an approximate weighted graph. Therefore, the clustering task is still performed on sample affinities, so higher-order affinity is not properly exploited to overcome the drawbacks of pairwise relations.

Recently, the use of tensors to encode high-order affinities has been studied in clustering. [24] introduce the characteristic tensor of a hypergraph and its associated Z-eigenvalue, providing natural relations for the structure of data. [25] denotes the order-k affinities by a symmetric tensor and relaxes the hypergraph clustering to solve the multilinear singular value decomposition problem. However, these methods can use only a specific order of affinity for clustering, which cannot completely express the data structure. Our early work of IPS2 [26] concentrates on using the high-order structural information of data that can be obtained from relationships between two pairs to enhance clustering performance. The authors establish an equivalence between the even-order affinity

[1]*School of Computer Science and Engineering, South China University of Technology, Guangzhou, 510006, China.* [2]*School of Data Science and Information Engineering, Guizhou Minzu University, China.* [3] *School of Computer Science, Guangdong Polytechnic Normal University, Guangzhou 510665, China.* [4] *School of Medical Technology, Beijing Institute of Technology, Beijing, China.* **Corresponding Author, E-mail: hmcai@scut.edu.cn, bh@bit.edu.cn.*

and the pairwise second-order affinity. However, there are still two problems that need to be solved: (1) the connection between odd-order tensor affinity and the second-order matrix affinity remains unknown; (2) how to jointly utilize affinities with different orders to accomplish clustering has yet to be studied.

To address these two questions, this paper first establishes a connection between the second-order and third-order tensor affinities via the Khatri-Rao product. Then, we propose a unified tensor clustering (UTC) model to learn a low-dimensional embedding by jointly using affinities of various orders. The popular second-order matrix affinity and the triadic and tetradic tensor affinities are fused to help characterize the proximity among samples.

The main advantages of the method proposed in this paper are summarized as follow.

1) A connection between pairwise matrix affinity and the triadic tensor affinity among three samples is established. An undecomposable third-order tensor is designed to provide the geometric proximity among three samples, thus serving as a supplement to the pairwise similarity matrix.
2) The third-order tensor affinity is defined via the Khatri-Rao product, while the Kronecker product is used to define the fourth-order tensor affinity. Therefore, the popular matrix affinity defined by the arithmetical product is surprisingly related to the Khatri-Rao and Kronecker products.
3) A uniform tensor clustering method (UTC) is developed. The UTC learns a low-dimensional embedding jointly based on affinities of various orders. The UTC is formulated elegantly and works on affinities of any order.
4) Extensive experiments on HDLSS data demonstrate that the UTC achieved superior performance compared with that of baseline methods. In addition, the experiments show that using high-order affinities to characterize the sample spatial distribution can improve clustering performance, especially in cases of small sample sizes.

2 NOTATIONS AND PRELIMINARIES

2.1 Notations

Throughout the paper, lowercase letters represent scalars, while bold lowercase letters represent vectors, such as $\boldsymbol{v} \in \mathbb{R}^m$. Bold uppercase letters denote matrices, such as $\boldsymbol{M} \in \mathbb{R}^{m\times n}$, while bold calligraphy letters represent tensors, such as, $\mathcal{T} \in \mathbb{R}^{m\times n\times l}$. Let $\mathcal{T}(:,:,i)$, $\mathcal{T}(:,i,:)$, $\mathcal{T}(i,:,:)$ represent the i-th frontal, lateral and horizontal slices of an order-3 tensor $\mathcal{T} \in \mathbb{R}^{n_1\times n_2\times n_3}$, respectively. $\mathcal{T}(:,:,i)$ can be abbreviated as $\mathcal{T}^{(i)}$. A tensor can be reformatted into a matrix via series operations, called *unfolding*. For example, the three-order and four-order tensor unfold operations are as follows:

Definition 2.1. ***Unfolding 3-order Tensor:*** *Let $\mathcal{T}_3 \in \mathbb{R}^{m\times m\times m}$ be an order-3 m-dimensional tensor. It can unfold to an $m^2 \times m$ matrix $\hat{\boldsymbol{T}}_3$ as follows:*

$$\hat{\boldsymbol{T}}_3 = \text{unfold}(\mathcal{T}_3) = \begin{bmatrix} \mathcal{T}_3^{(1)} \\ \mathcal{T}_3^{(2)} \\ \vdots \\ \mathcal{T}_3^{(m)} \end{bmatrix} \tag{1}$$

Definition 2.2. ***Unfolding 4-order Tensor*** *Let $\mathcal{T}_4 \in \mathbb{R}^{m\times m\times m\times m}$ represent a fourth-order m-dimensional tensor. This tensor can unfold to an $m^2 \times m^2$ matrix $\hat{T}_4$, with its (r,s)-th entry given by:*

$$\hat{\boldsymbol{T}}_{4_{rs}} = \mathcal{T}_{4_{ijkl}} \tag{2}$$

with $r = m(j-1)+i, s = m(l-1)+k, i,j,k,l \in [1,m]$.

Three matrix multiplications are considered, namely, Hadamard product, Kronecker product and Khatri-Rao product [27].

Definition 2.3. ***Hadamard Product*** *The Hadamard product of two matrices $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{m\times n}$ is defined as:*

$$\boldsymbol{A} \odot \boldsymbol{B} = \begin{bmatrix} a_{11}b_{11} & \cdots & a_{1n}b_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1}b_{m1} & \cdots & a_{mn}b_{mn} \end{bmatrix} \in \mathbb{R}^{m\times n} \tag{3}$$

Definition 2.4. ***Kronecker Product*** *The Kronecker product of two matrices $\boldsymbol{A} \in \mathbb{R}^{m_1\times n_1}$ and $\boldsymbol{B} \in \mathbb{R}^{m_2\times n_2}$ is defined as:*

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & \cdots & a_{1n}\boldsymbol{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\boldsymbol{B} & \cdots & a_{mn}\boldsymbol{B} \end{bmatrix} \in \mathbb{R}^{m_1m_2\times n_1n_2} \tag{4}$$

Definition 2.5. ***Khatri-rao Product*** *The Khatri-Rao product of two matrices $\boldsymbol{A} \in \mathbb{R}^{m_1\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{m_2\times n}$ is defined as the matrix:*

$$\boldsymbol{A} * \boldsymbol{B} = \begin{bmatrix} a_{:1} \otimes b_{:1} & \cdots & a_{:n} \otimes b_{:n} \end{bmatrix} \in \mathbb{R}^{m_1m_2\times n} \tag{5}$$

A popular product between a tensor and a matrix is called a *k-mode product*.

Definition 2.6. ***k-mode Product*** *The k-mode product between an order-m tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_m}$ and a matrix $V \in \mathbb{R}^{p\times n_k}$, denoted by $\boldsymbol{\mathcal{T}} \otimes_k U \in \mathbb{R}^{n_1\times\cdots\times n_{k-1}\times p\times n_{k+1}\times\cdots\times n_m}$, with*

$$(\boldsymbol{\mathcal{T}} \otimes_k \boldsymbol{V})_{i_1\ldots i_{k-1}ji_{k+1}\ldots i_m} = \sum_{i_k=1}^{n_k} \boldsymbol{\mathcal{T}}_{i_1\ldots i_{k-1}i_ki_{k+1}\ldots i_m} \boldsymbol{V}_{ji_k} \tag{6}$$

For convenience, we summarize the frequently used notations and definitions in Table 1.

TABLE 1: Summary of notations

$\boldsymbol{v} \in \mathbb{R}^m$	A vector
$\boldsymbol{M} \in \mathbb{R}^{m\times n}$	A matrix
$\mathcal{T} \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_k}$	An order-k tensor
$\hat{\boldsymbol{T}}_k$	The unfolding order-k tensor
$Tr(\cdot)$	The trace operation
$\odot$	The Hadamard product
$*$	The Khatri-Rao product
$\otimes$	The Kronecker product
$\otimes_k$	The k-mode product

2.2 Introduction of Spectral Clustering with Pairwise Similarity

Spectral clustering [3] starts by computing a sample-wise affinity matrix $\boldsymbol{S}_2 \in \mathbb{R}^{m\times m}$. Every entry $\boldsymbol{S}_2(i,j)$ measures the proximity between two samples x_i and x_j. Then, a normalized affinity matrix $\hat{\boldsymbol{L}}_2$ is estimated with $\hat{\boldsymbol{L}}_2 = \boldsymbol{D}_2^{-\frac{1}{2}}\boldsymbol{S}_2\boldsymbol{D}_2^{-\frac{1}{2}}$, where $\boldsymbol{D}_2$ denotes the degree matrix of $\boldsymbol{S}_2$ with $\boldsymbol{D}_{2_{ii}} = \sum_i \boldsymbol{S}_{2ij}$. The Laplacian matrix $\boldsymbol{L}$ can be generated through $\boldsymbol{L} = \boldsymbol{I} - \hat{\boldsymbol{L}}_2$.

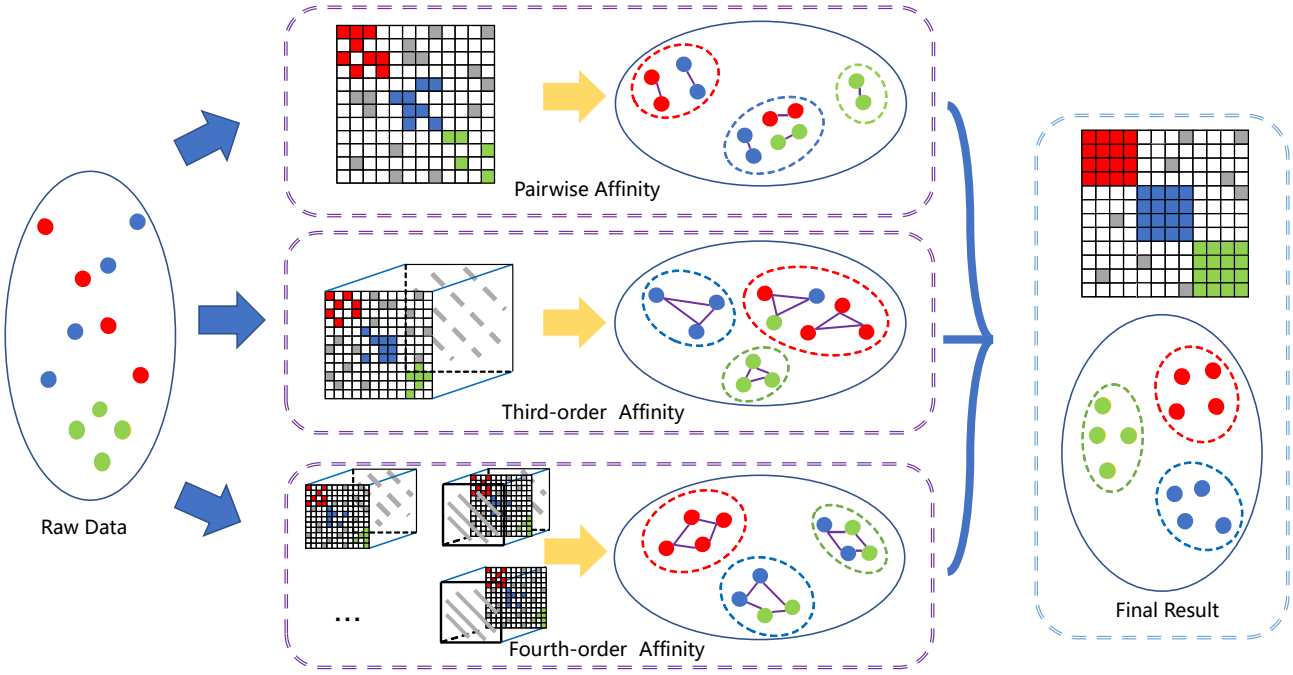


Fig. 1: Demonstration of the UTC method, which aims to learn a uniform embedding via joint optimization of not only the pairwise matrix affinity but also the triadic and tetradic tensor affinities among more than two samples. The approach is combined with the high-order tensor affinity to provide complementary spatial characterization for which the pairwise affinity fails; UTC learns a low-dimensional embedding and is robust to high-dimensional concentration effects and remains stable.

Spectral graph clustering seeks a low-dimensional embedding of samples by maximizing the cross-entropy while preserving the local proximity of paired samples:

$$\begin{aligned} &\min_{\boldsymbol{V}} \ tr(\boldsymbol{V}^T \boldsymbol{L} \boldsymbol{V}) \\ \text{Subject to:} \quad &\boldsymbol{V}^T \boldsymbol{V} = \boldsymbol{I} \end{aligned} \tag{7}$$

It is a standard eigenvalue problem and thus can be efficiently solved by calculating the eigenvectors $\boldsymbol{V}$ of the Laplacian matrix $\boldsymbol{L}$, using methods such as singular value decomposition. Typically, the normalized cut of the graph is minimized in this model, but as stated in [28], this model can also be expressed as a maximization problem with the normalized pairwise affinity $\hat{\boldsymbol{L}}_2$. By using the tensor k-mode product, the spectral clustering in Eq. (7) can be rewritten as:

$$\begin{aligned} &\max_{\boldsymbol{V}} \ tr(\boldsymbol{V}^T \hat{\boldsymbol{L}}_2 \boldsymbol{V}) \\ =&\max_{\boldsymbol{V}} \ \sum_{j=1}^{k} \hat{\boldsymbol{L}}_2 \otimes_2 \boldsymbol{v}_j^T \otimes_1 \boldsymbol{v}_j^T \\ &\text{Subject to: } \boldsymbol{V}^T \boldsymbol{V} = \boldsymbol{I} \end{aligned} \tag{8}$$

where $\boldsymbol{v}_j$ is the j^{th} column of the partition matrix $\boldsymbol{V}$. This equation establishes the relationship between the membership $\boldsymbol{v}_j$ and the pairwise affinity. Then, a clustering task is performed on the obtained embedding.

2.3 Introduction of IPS2 Clustering with Pair-to-Pair Similarity

The pairwise affinity is easily broken by noise contamination or concentration effects in HDLSS data. To address this issue, our recent work IPS2 [26] attempted to use high-order affinity among more than two samples. IPS2 used the fourth-order tensor affinity to measure the proximity of two sample pairs. The decomposable fourth-order tensor affinity can be defined as the product of two pairwise similarities:

$$\mathcal{T}_{4_{ijkl}} = \boldsymbol{S}_{2_{ik}} \boldsymbol{S}_{2_{jl}}, \quad i,j,k,l \in m. \tag{9}$$

Let $\hat{\boldsymbol{T}}_4$ denote the matrix unfolded from a fourth-order decomposable tensor affinity $\mathcal{T}_4$; it has been shown in [26] that

$$\hat{\boldsymbol{T}}_4 = \boldsymbol{S}_2 \otimes \boldsymbol{S}_2 \tag{10}$$

This result establishes a direct connection between a fourth-order decomposable tensor affinity and the usual second-order matrix affinity $\boldsymbol{S}_2$. Now, consider a normalized affinity matrix $\hat{\boldsymbol{L}}_4$, computed by $\hat{\boldsymbol{L}}_4 = \hat{\boldsymbol{D}}_4^{-\frac{1}{2}} \hat{\boldsymbol{T}}_4 \hat{\boldsymbol{D}}_4^{-\frac{1}{2}}$. Here, the diagonal matrix $\hat{\boldsymbol{D}}_4$ is the degree matrix of $\hat{\boldsymbol{T}}_4$ with the diagonal elements being $\hat{\boldsymbol{D}}_{4_{ii}} = \sum_j \hat{\boldsymbol{T}}_{4_{ij}}$. It has been shown that the fourth-order and second-order normalized affinity matrix $\hat{\boldsymbol{L}}_4, \hat{\boldsymbol{L}}_2$ shares a similar relation as in Eq. (10):

Theorem 2.1. ***The decomposable 4-order tensor [26]*** *Let $\hat{\boldsymbol{L}}_2$ and $\mathcal{L}_4$ denote the normalized similarity matrix and fourth-order normalized affinity tensor with the unfolding form $\hat{\boldsymbol{L}}_4$, respectively. Then, we have the following equality:*

$$\hat{\boldsymbol{L}}_4 = \hat{\boldsymbol{L}}_2 \otimes \hat{\boldsymbol{L}}_2 \tag{11}$$

Moreover, the eigenvectors $\boldsymbol{v}$ and $\hat{\boldsymbol{v}}$ for the matrix $\hat{\boldsymbol{L}}_2$ and unfolded matrix $\hat{\boldsymbol{L}}_4$ satisfy:

$$\hat{\boldsymbol{v}} = \boldsymbol{v} \otimes \boldsymbol{v} \tag{12}$$

One can reshape the eigenvector $\hat{\boldsymbol{v}}$ into a matrix $\boldsymbol{V} \in \mathbb{R}^{m \times m}$, and the matrix $\boldsymbol{V}$ is an eigenmatrix of the fourth-order tensor

affinity $\hat{\boldsymbol{T}}_4$ in Eq. (10). The established equivalence between the decomposable tensor affinity $\mathcal{T}_4$ and any similarity matrix $\boldsymbol{S}$ opens a new path to understand high-order affinity. However, the fourth-order decomposable affinity inherits the drawback of the low-order similarity [29] in that it is vulnerable to noise corruption and concentration effects when applied to high-dimensional samples. Alternatively, one can use undecomposable tensor affinity to provide complementary sample affinities that the pairwise matrix affinity could not provide. The IPS2 uses a Fisher-ratio-like tensor affinity:

$$\mathcal{T}_{ijkl} = exp(-\sigma \frac{d_{ij} + d_{kl}}{d_{ik} + d_{jl} + \varepsilon}) \tag{13}$$

for $i, j, k, l \in n$, where d_{ij} denotes the distance between samples x_i and x_j and the parameter σ is an scaling constant.

One can calculate the normalized tensor affinity $\mathcal{L}_4$ from the fourth-order tensor affinity $\mathcal{T}_4$. Then, we can learn a low-dimensional embedding $\boldsymbol{V}$ from the fourth-order tensor affinity by solving the following maximization problem:

$$\begin{aligned} \max_{\boldsymbol{V}} & \sum_{j=1}^{k} \mathcal{L}_4 \otimes_4 \boldsymbol{v}_j \otimes_3 \boldsymbol{v}_j \otimes_2 \boldsymbol{v}_j \otimes_1 \boldsymbol{v}_j \\ & = tr((\boldsymbol{V} * \boldsymbol{V})^T \hat{\boldsymbol{L}}_4 (\boldsymbol{V} * \boldsymbol{V})) \\ & s.t. \boldsymbol{V}^T \boldsymbol{V} = \boldsymbol{I} \end{aligned} \tag{14}$$

3 METHOD

The performance of clustering based on pairwise affinity suffers in high-dimensional and small-sample-size datasets. Incorporating the spatial distribution among multiple samples instead of using only pairwise affinity improves the clustering performance. IPS2 [26] built a bridge between the fourth-order affinity and low-dimensional embedding. It has been shown that high-order information can effectively complete the description of spatial data structure from low-order information. However, it is restricted to using even-order affinity, such as the fourth-order affinity, and does not apply to odd-order affinity. Moreover, it fails to fuse affinities from different orders within a uniform formulation, resulting in cumbersome utility for empirical applications. To address these two issues, we did the following work: (1) we introduced triadic and tetradic tensor affinity for three and four samples; (2) we propose a uniform framework to learn a low-dimensional embedding by fusing affinity of various orders. An illustrative overview of UTC is shown in Fig. 1.

3.1 Characterizing Sample's Affinity via Third-Order Tensor Affinity

3.1.1 Decomposable Third-Order Tensor Affinity

Popular clustering methods are based on the pairwise similarity of two samples. We are interested in the tensor affinity among three samples, denoted by a third-order tensor $\mathcal{T}_3 = [\mathcal{T}_{ijk}] \in \mathbb{R}^{m \times m \times m}$. A natural way to define such affinity is to use the composite affinity based on paired affinities, such that each entry $\mathcal{T}_{ijk}$ is given by:

$$\mathcal{T}_{3_{ijk}} = \boldsymbol{S}_{2_{ij}} \boldsymbol{S}_{2_{kj}} \tag{15}$$

Under this definition, one can generalize the pairwise affinity $\boldsymbol{S}_2$ to a third-order tensor affinity $\mathcal{T}_3$. Specifically, the results satisfy the following properties.

Theorem 3.1. *Given a decomposable third-order tensor affinity $\mathcal{T}_3$ defined in Eq. (15), $\hat{\boldsymbol{T}}_3 = \boldsymbol{S}_2 * \boldsymbol{S}_2$, where $\hat{\boldsymbol{T}}_3$ is the matrix unfolded by the tensor $\mathcal{T}_3$.*

Proof. By the definition in Eq.(1), the unfolded matrix $\hat{\boldsymbol{T}}_3 \in \mathbb{R}^{n^2 \times n}$ can be written:

$$\begin{aligned} \hat{\boldsymbol{T}}_3 &= \begin{bmatrix} \mathcal{T}_{3_{111}} & \dots & \mathcal{T}_{3_{1n1}} \\ \vdots & \ddots & \vdots \\ \mathcal{T}_{3_{n1n}} & \dots & \mathcal{T}_{3_{nnn}} \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{S}_{2_{11}}\boldsymbol{S}_{2_{11}} & \dots & \boldsymbol{S}_{2_{1n}}\boldsymbol{S}_{2_{1n}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{S}_{2_{11}}\boldsymbol{S}_{2_{n1}} & \dots & \boldsymbol{S}_{2_{1n}}\boldsymbol{S}_{2_{nn}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{S}_{2_{n1}}\boldsymbol{S}_{2_{11}} & \dots & \boldsymbol{S}_{2_{nn}}\boldsymbol{S}_{2_{1n}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{S}_{2_{n1}}\boldsymbol{S}_{2_{n1}} & \dots & \boldsymbol{S}_{2_{nn}}\boldsymbol{S}_{2_{nn}} \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{S}_{2_{11}}\boldsymbol{S}_{2_{:1}} & \dots & \boldsymbol{S}_{2_{1n}}\boldsymbol{S}_{2_{:n}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{S}_{2_{n1}}\boldsymbol{S}_{2_{:1}} & \dots & \boldsymbol{S}_{2_{nn}}\boldsymbol{S}_{2_{:n}} \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{S}_{2_{:1}} \otimes \boldsymbol{S}_{2_{:1}} & \dots & \boldsymbol{S}_{2_{:n}} \otimes \boldsymbol{S}_{2_{:n}} \end{bmatrix} \\ &= \boldsymbol{S}_2 * \boldsymbol{S}_2 \end{aligned} \tag{16}$$

□

Surprisingly, this theorem shows that a decomposable third-order tensor affinity can be obtained via the Khatri-Rao product of two affinity matrices. We now proceed to define a normalized affinity tensor $\mathcal{L}_3$. Let a matrix $\hat{\boldsymbol{L}}_3$ be calculated by $\hat{\boldsymbol{L}}_3 = \hat{\boldsymbol{D}}_{3_1}^{-\frac{1}{2}} \hat{\boldsymbol{T}}_3 \hat{\boldsymbol{D}}_{3_2}^{-\frac{1}{2}}$. Here, the two diagonal matrices $\hat{\boldsymbol{D}}_{3_1}, \hat{\boldsymbol{D}}_{3_2}$ are given by $\hat{\boldsymbol{D}}_{3_1} = \sqrt{\sum_i \hat{\boldsymbol{T}}_{3_{:i}} \sum_i \hat{\boldsymbol{T}}_{3_{:i}}} = \boldsymbol{D}_2 \otimes \boldsymbol{D}_2$ and $\hat{\boldsymbol{D}}_{3_2} = \sum_i \hat{\boldsymbol{T}}_{3_{:i}} = \boldsymbol{D}_2 \odot \boldsymbol{D}_2$. The normalized third-order affinity tensor $\mathcal{L}_3$ is obtained by folding the matrix $\hat{\boldsymbol{L}}_3$.

This defined affinity tensor shares a similar property to that of the fourth-order affinity tensor.

Theorem 3.2. *Let a matrix $\boldsymbol{S}_2$ be a pairwise affinity matrix with $\hat{\boldsymbol{L}}_2$ being its normalized affinity matrix. The decomposable third-order tensor affinity $\mathcal{T}_3$ and its normalized tensor $\mathcal{L}_3$ are obtained as above. We have*

$$\hat{\boldsymbol{L}}_3 = \hat{\boldsymbol{L}}_2 * \hat{\boldsymbol{L}}_2 \tag{17}$$

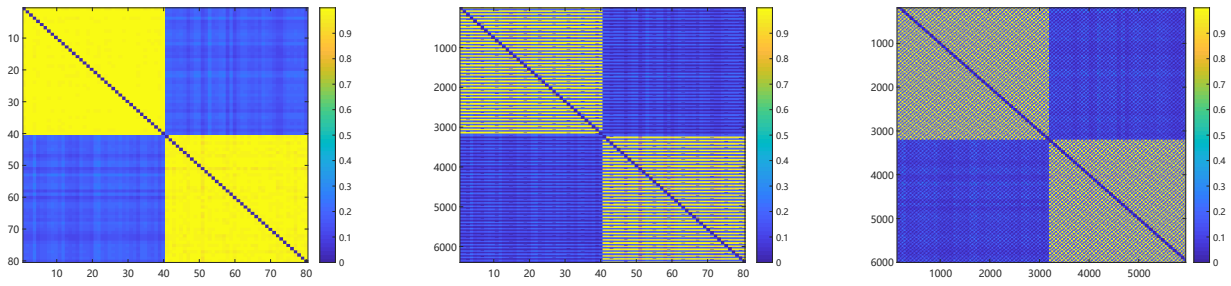
where $\hat{\boldsymbol{L}}_3$ is the unfolded matrix from the tensor $\mathcal{L}_3$.

Proof. By the definitions of the normalized affinity matrix $\hat{\boldsymbol{L}}_2$ and $\hat{\boldsymbol{L}}_3$, we have:

$$\begin{aligned} \hat{\boldsymbol{L}}_3 =& \hat{\boldsymbol{D}}_{3_1}^{-\frac{1}{2}} \hat{\boldsymbol{T}}_3 \hat{\boldsymbol{D}}_{3_2}^{-\frac{1}{2}} \\ =& (\boldsymbol{D}_2 \otimes \boldsymbol{D}_2)^{-\frac{1}{2}} (\boldsymbol{S}_2 * \boldsymbol{S}_2)(\boldsymbol{D}_2 \odot \boldsymbol{D}_2)^{-\frac{1}{2}} \\ =& (\boldsymbol{D}_2^{-\frac{1}{2}} \boldsymbol{S}_2 \boldsymbol{D}_2^{-\frac{1}{2}}) * (\boldsymbol{D}_2^{-\frac{1}{2}} \boldsymbol{S}_2 \boldsymbol{D}_2^{-\frac{1}{2}}) \\ =& \hat{\boldsymbol{L}}_2 * \hat{\boldsymbol{L}}_2 \end{aligned} \tag{18}$$

□

Theo. 3.1 and Theo. 3.2 build a connection between third-order affinity $\mathcal{T}_3$ and pairwise similarity $\boldsymbol{S}_2$ via the Khatri-Rao product. We construct a synthetic dataset consisting of eighty samples, evenly drawn from two different clusters. Then, we visualize the



(a) (b) (c)

Fig. 2: Demonstration of (a) pairwise, (b) third-order, and (c) fourth-order affinity on a synthetic dataset drawn from a normal distribution. There are obvious block structures in the heatmap on different orders of unfolded affinity, indicating that the decomposable similarities of different orders are consistent in terms of performance.

pairwise affinity $\boldsymbol{S}_2$ and the decomposable third-order and fourth-order tensor affinity in Fig. 2. One can observe that they preserve the sample proximity satisfactorily. They can be well segmented into four distinct blocks, and within each sub-block, the sample proximity is small.

3.1.2 Indecomposable Third-Order Tensor Affinity

After constructing the connection of decomposable affinity with pairwise similarity, we analyze the local spatial relations to obtain supplement information for pairwise relations. As [30] stated, when the order is two, the spectral methods can also be seen as an eigenvector $\boldsymbol{v}$ containing the important underlying manifold information of the data. We generate this model to high-order affinity. Different third-order information can be extracted by examining the three samples from different views. As an example, the affinity between three samples is defined as follows: for any three samples x_i, x_j, and x_k, we can always use one sample x_j as an anchor point for the other samples x_i and x_k and consider anchor point x_j's perspective to measure the relationship between the two other points. If x_i and x_j are sufficiently similar, they are similar no matter the location of the anchor point. But for outliers, the similarity to other data should be small under most anchors' perspectives. Therefore, the undecomposable affinity $\mathcal{T}_3 \in \mathbb{R}^{m^2 \times m}$ among three points is defined by:

$$\mathcal{T}_{3_{ijk}} = \frac{< (x_i - x_j), (x_k - x_j) >}{d_{ij} d_{jk}} \tag{19}$$

for $i, j, k \in m$, where d_{ij} denotes the distance between samples x_i and x_j. The entry $\mathcal{T}_{3_{ijk}}$ denotes the similarity between x_i and x_k under x_j's view. If the distance between x_i and x_k is small, wherever the location of x_j is, it should be a small value. Then, we can unfold $\mathcal{T}_3$ into a matrix $\hat{\boldsymbol{T}}_3 \in \mathbb{R}^{m^2 \times m}$ along the mode-3 direction. Similar to the decomposable tensor affinity, we can infer the three-order normalized affinity tensor $\mathcal{L}_3$, which when unfolded along the mode-3 direction satisfies $\hat{\boldsymbol{L}}_3 = \hat{\boldsymbol{D}}_{3_1}^{-\frac{1}{2}} \hat{\boldsymbol{T}}_3 \hat{\boldsymbol{D}}_{3_2}^{-\frac{1}{2}}$, where $\hat{\boldsymbol{D}}_{3_1}^{-\frac{1}{2}} = \sqrt{\sum_i \hat{\boldsymbol{T}}_{3_{::i}} \sum_i \hat{\boldsymbol{T}}_{3_{::i}}} = \boldsymbol{D}_2 \otimes \boldsymbol{D}_2$ and $\hat{\boldsymbol{D}}_{3_2} = \sum_i \hat{\boldsymbol{T}}_{3_{::i}} = \boldsymbol{D}_2 * \boldsymbol{D}_2$. Similar to the form of the pairwise and tetradic relations, we seek a low-dimensional embedding $\boldsymbol{v}$ from the third-order tensor affinity by maximizing the cross-entropy:

$$\begin{aligned} &\max_{\boldsymbol{V}} \sum_{j=1}^{k} \mathcal{L}_3 \otimes_3 \boldsymbol{v}_j \otimes_2 \boldsymbol{v}_j \otimes_1 \boldsymbol{v}_j \\ &= \max_{\boldsymbol{V}} \ tr((\boldsymbol{V} * \boldsymbol{V})^T \hat{\boldsymbol{L}}_3 \boldsymbol{V}) \\ &\quad s.t. \boldsymbol{V}^T \boldsymbol{V} = \boldsymbol{I} \end{aligned} \tag{20}$$

3.2 Uniform Tensor Clustering

We now formulate a uniform framework to learn a low-dimensional embedding from affinities of various orders. Suppose we have n samples $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_m] \in \mathbb{R}^{n \times m}$. The aim of clustering is to assign the samples into k disjoint clusters, $C_1, C_2, \ldots, C_k$. We introduce an entropy, called total normalized similarity, to quantify the sample assignment:

Definition 3.1. ***The total N-similarity of a cluster:*** *Let $\boldsymbol{C} \in \boldsymbol{X}$ be a group of samples and $\mathcal{S} \in \mathbb{R}^{n_1 \times n_2 \cdots \times n_k}$ be the order-k similarity tensor. The total normalized similarity of $\boldsymbol{C}$ is defined as*

$$Sim(\boldsymbol{C}) = \sum_{x_{i_1}, x_{i_2}, \ldots, x_{i_k} \in \boldsymbol{C}} \mathcal{L}_{x_{i_1}, x_{i_2}, \ldots, x_{i_k}} \tag{21}$$

where $\mathcal{L}$ is the normalized affinity tensor calculated by $\mathcal{S}$. In addition, for a partition $\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_k$ of $\boldsymbol{X}$, we define the normalized associativity of the clustering as:

$$N - Assoc(\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_k) = \sum_{i=1}^{k} \frac{Sim(\boldsymbol{C}_i)}{|\boldsymbol{C}_j|^m}, \tag{22}$$

where $|\boldsymbol{C}_j|$ denotes the sample number of cluster $\boldsymbol{C}_j$.

Let an indicator matrix $\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_k] \in \mathbb{R}^{n \times k}$ be the sample assignment such that $\boldsymbol{H}_{ij} = |\boldsymbol{C}_j|^{-1}$ if $x_i \in \boldsymbol{C}_j$ and zero otherwise.

One can maximize the normalized associativity in Eq. (22) to seek an optimal sample assignment $\boldsymbol{C}_i, \ldots, \boldsymbol{C}_k$. Via algebraic transformation, this problem is equivalent to solving:

$$\max_{\boldsymbol{C}_1, \cdots, \boldsymbol{C}_k} \sum_{j=1}^{k} (\mathcal{L} \otimes_m \boldsymbol{h}_j \otimes_{m-1} \boldsymbol{h}_j \cdots \otimes_1 \boldsymbol{h}_j) \tag{23}$$

where $\boldsymbol{h}_j$ denotes the j^{th} column of $\boldsymbol{H}$.

One can verify that solving the maximum normalized associativity is NP-hard. Alternatively, one can relax the binary assignment matrix to the orthonormal matrix $\boldsymbol{V} \in \mathbb{R}^{n \times k} : \boldsymbol{V}^T \boldsymbol{V} = \boldsymbol{I}$

to lower the expectation of binary assignment. The simplified problem then becomes solving

$$\max_{\boldsymbol{V}^T\boldsymbol{V}=\boldsymbol{I}} \sum_{j=1}^{k} \mathcal{L} \otimes_1 \boldsymbol{v}_j \otimes_2 \boldsymbol{v}_j \cdots \otimes_m \boldsymbol{v}_j \tag{24}$$

where $\boldsymbol{v}_j$ represents the j^{th} column of $\boldsymbol{V}$.

We now fuse the aforementioned undecomposable third-order and fourth-order tensor affinities with the pairwise affinity matrix to seek a uniform low-dimensional embedding. The high-order tensor affinity can be used to effectively address the challenge of the vulnerability to high-dimensional datasets. The proposed model, named uniform tensor clustering (UTC), effectively fuses the similarity measurements among samples to yield a uniform low-dimensional embedding to approximate the partition matrix. It achieves the goal by maximizing the cluster's associativity of various orders. The proposed model is formulated as:

$$\begin{aligned} \min_{\boldsymbol{V}} \ & \sum_{j=1}^{k} -\hat{L}_2 \otimes_2 \boldsymbol{v}_j \otimes_1 \boldsymbol{v}_j - \mathcal{L}_3 \otimes_3 \boldsymbol{v}_j \otimes_2 \boldsymbol{v}_j \otimes_1 \boldsymbol{v}_j \\ & - \mathcal{L}_4 \otimes_4 \boldsymbol{v}_j \otimes_3 \boldsymbol{v}_j \otimes_2 \boldsymbol{v}_j \otimes_1 \boldsymbol{v}_j \\ & s.t\ \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I} \end{aligned} \tag{25}$$

where $\boldsymbol{v}_j$ is the j^{th} column of the matrix $\boldsymbol{V}$. This unified model aims to fuse various affinity matrices to yield a uniform low-dimensional embedding that is robust to noise contamination and high-dimensional concentration effects. The proposed model enhances the clustering performance by exploiting spatial information from multiple (more than two) samples. Upon solving this problem, one obtains the embedding $\boldsymbol{V}$. The clustering task is accomplished by fusing low- to high-order information and using K-means clustering based on the embedding matrix $\boldsymbol{V}$ to obtain the final group. Therefore, once we have the indecomposable tensor affinity $\mathcal{S}$ from different orders, computing the fused low-dimensional embedding yields the different orders' spatial information to enhance the clustering performance.

3.3 Numerical Scheme to Solve UTC

The optimization problem in Eq. (25) is convex, and the objective is differentiable. However, the gradient of the objective function involves solving a polynomial function of order three. It has no explicit solution and thus is computationally intractable. To circumvent this issue, we introduce a slack variable $\boldsymbol{v}_2$ to approximate the term $\boldsymbol{v} \otimes \boldsymbol{v}$, thereby avoiding having to solve a high-order polynomial function. The model can be rewritten as

$$\begin{aligned} \min_{\boldsymbol{V}_1,\boldsymbol{V}_2} \ & - tr(\boldsymbol{V}_1^T\hat{\boldsymbol{L}}_2\boldsymbol{V}_1 + \boldsymbol{V}_2^T\hat{\boldsymbol{L}}_3\boldsymbol{V}_1 + \boldsymbol{V}_2^T\hat{\boldsymbol{L}}_4\boldsymbol{V}_2) \\ & s.t.\ \boldsymbol{V}_1^T\boldsymbol{V}_1 = \boldsymbol{I} \\ & \quad \boldsymbol{V}_1 * \boldsymbol{V}_1 = \boldsymbol{V}_2 \end{aligned} \tag{26}$$

This problem can be solved by alternating direction minimization. Its augmented Lagrangian formulation is as follows:

$$\begin{aligned} \min_{\boldsymbol{V}_1,\boldsymbol{V}_2} \ & - tr(\boldsymbol{V}_1^T\hat{\boldsymbol{L}}_2\boldsymbol{V}_1) - tr(\boldsymbol{V}_2^T\hat{\boldsymbol{L}}_3\boldsymbol{V}_1) - tr(\boldsymbol{V}_2^T\hat{\boldsymbol{L}}_4\boldsymbol{V}_2) \\ & + < \boldsymbol{Y}_1, \boldsymbol{V}_1 * \boldsymbol{V}_1 - \boldsymbol{V}_2 > + < \boldsymbol{Y}_2, \boldsymbol{V}_1^T\boldsymbol{V}_1 - \boldsymbol{I} > \\ & + \frac{\mu}{2}[\|\boldsymbol{V}_1 * \boldsymbol{V}_1 - \boldsymbol{V}_2\|_F^2 + \|\boldsymbol{V}_1^T\boldsymbol{V}_1 - \boldsymbol{I}\|_F^2] \end{aligned} \tag{27}$$

where $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ are Lagrange multipliers. The constant $\mu > 0$ is a penalty parameter. We then decompose the problem into two subproblems with respect to the variables $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$ alternatively. Each variable is solved while fixing other variables. The process is updated iteratively until convergence.

Step 1): Solving the subproblem with respect to the variable $\boldsymbol{V}_1$

By fixing the variable $\boldsymbol{V}_2$, the problem can be simplified as:

$$\begin{aligned} \min_{\boldsymbol{V}_1} \ & - tr(\boldsymbol{V}_1^T\boldsymbol{L}\boldsymbol{V}_1) - tr(\boldsymbol{V}_2^T\hat{\boldsymbol{L}}_3\boldsymbol{V}_1) \\ & + < \boldsymbol{Y}_1, \boldsymbol{V}_1 * \boldsymbol{V}_1 - \boldsymbol{V}_2 > + < \boldsymbol{Y}_2, \boldsymbol{V}^T\boldsymbol{V} - \boldsymbol{I} > \\ & + \frac{\mu}{2}[\|\boldsymbol{V}_1 * \boldsymbol{V}_1 - \boldsymbol{V}_2\|_F^2 + (\boldsymbol{V}^T\boldsymbol{V} - \boldsymbol{I})] \end{aligned} \tag{28}$$

The gradient of the quadratic function is:

$$\begin{aligned} \frac{\partial L}{\partial \boldsymbol{V}_1} = & -2\boldsymbol{L}\boldsymbol{V}_1 - \hat{\boldsymbol{L}}_3^T\boldsymbol{V}_2 + \mu\sum_{j=1}^{k}(\boldsymbol{V}_{1:_j} \otimes \boldsymbol{I} \\ & + \boldsymbol{I} \otimes \boldsymbol{V}_{1:_j})^T(\boldsymbol{V}_{1:_j} * \boldsymbol{V}_{1:_j} - \boldsymbol{V}_{2:_j} + \boldsymbol{Y}_{1:_j}/\mu) \\ & + 2\mu\boldsymbol{V}_1(\boldsymbol{V}_1^T\boldsymbol{V}_1 - \boldsymbol{I} + \boldsymbol{Y}_2/\mu) \end{aligned} \tag{29}$$

Thus, the problem in Eq. (28) can be solved via gradient descent or by setting the gradient function Eq. (29) to be zero to obtain the solution directly.

Step 2): Solving the subproblem with respect to the variable $\boldsymbol{V}_2$

By fixing the variable $\boldsymbol{V}_1$, the augmented Lagrange function can be simplified as:

$$\begin{aligned} \min_{\boldsymbol{V}_2} \ & tr(-\boldsymbol{V}_2^T\hat{\boldsymbol{L}}_3\boldsymbol{V}_1) - tr(\boldsymbol{V}_2^T\hat{\boldsymbol{L}}_4\boldsymbol{V}_2) \\ & + \boldsymbol{Y}_1(\boldsymbol{V}_1 * \boldsymbol{V}_1 - \boldsymbol{V}_2) + \frac{\mu}{2}[\|\boldsymbol{V}_1 * \boldsymbol{V}_1 - \boldsymbol{V}_2\|_F^2 \end{aligned} \tag{30}$$

The gradient of the objective function is:

$$\frac{\partial L}{\partial \boldsymbol{V}_2} = -\hat{\boldsymbol{L}}_3\boldsymbol{V}_1 - 2\hat{\boldsymbol{L}}_4\boldsymbol{V}_2 + \mu(\boldsymbol{V}_2 - \boldsymbol{V}_1 * \boldsymbol{V}_1 - \frac{\boldsymbol{Y}_1}{\mu}) \tag{31}$$

By setting the gradient to zero, one can obtain the implicit solution as:

$$\boldsymbol{V}_2^* = (\mu\boldsymbol{I} - 2\hat{\boldsymbol{L}}_4)^{-1}(\mu\boldsymbol{V}_1 * \boldsymbol{V}_1 + \hat{\boldsymbol{L}}_3\boldsymbol{V}_1 + \boldsymbol{Y}) \tag{32}$$

Step 3): Updating the multipliers $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$:

$$\boldsymbol{Y}_1^{(k+1)} = \boldsymbol{Y}_1^{(k)} + \mu(\boldsymbol{V}_1^{(k)} * \boldsymbol{V}_1^{(k)} - \boldsymbol{V}_2^{(k)}) \tag{33}$$

$$\boldsymbol{Y}_2^{k+1} = \boldsymbol{Y}_2^{(k)} + \mu({\boldsymbol{V}_1^{(k)}}^T\boldsymbol{V}^{(k)} - \boldsymbol{I}) \tag{34}$$

The three steps are iteratively updated until convergence or until a stopping criterion is met: $max(\|\boldsymbol{v}_1^{k+1} - \boldsymbol{v}_1^k\|_\infty, \|\boldsymbol{v}_2^{k+1} - \boldsymbol{v}_2^k\|_\infty, \|\boldsymbol{v}_1^{k+1} \otimes \boldsymbol{v}_1^{k+1} - \boldsymbol{v}_2^{k+1}\|_\infty) < \epsilon$. Since the main problem is convex, the numerical scheme is guaranteed to be convergent. The alternate strategy to solve the problem is summarized in Algorithm 1.

4 EXPERIMENTS

In this section, we illustrate the superiority of the proposed UTC method on both synthetic and real datasets. We first construct an example (Syndata-1) to demonstrate the scenario in which clustering through the standard pairwise affinity fails. In comparison, the triadic affinity can be employed to adequately characterize the spatial distribution of multiple samples; thus, the proposed UTC can perfectly cluster the sample using the triadic affinity. We then construct a second synthetic dataset (Syndata-2) with the HDLSS property to demonstrate the scenario in which clustering through

Algorithm 1 High-Order Affinity Clustering

Input: Data $\mathcal{X} \in \mathbb{R}^{m*d}$, Cluster number c

Output: The fused low-dimensional embedding $\boldsymbol{V}$ and the cluster results $\boldsymbol{Y}_{pred}$

Compute the pairwise affinity $\hat{\boldsymbol{L}}_2$. Set $\boldsymbol{V}_1^0 = \boldsymbol{V}_2^0 = \boldsymbol{Y}_1^0 = \boldsymbol{Y}_2^0 = \boldsymbol{0}$, $\mu^0 = 10^{-3}$, $\mu^{max} = 10^2$, $\epsilon = 10^{-2}$, $\varepsilon = 10^{-3}$, $\rho = 1.1$, $\alpha = 10^{-3}$, $k = t = 0$, and $\sigma = 10^{-6}$.
Construct triadic affinity tensor $\mathcal{L}_3$ by Eq. (19).
Construct tetradic affinity tensor $\mathcal{L}_4$ by Eq. (13).
Unfold order-3 tensor $\mathcal{L}_3$ to matrix $\hat{\boldsymbol{L}}_3$ by Eq. (1).
Unfold order-4 tensor $\mathcal{L}_4$ to matrix $\hat{\boldsymbol{L}}_4$ by Eq. (2).
while Not Converged **do**
  $\boldsymbol{V}_{1_{temp}}^t = \boldsymbol{V}_1^k$.
  while Not Converged **do**
    Fix $\boldsymbol{V}_2^k$, computed $\frac{\partial L}{\partial \boldsymbol{V}_{1_{temp}}^t}$ by Eq. (29).
    Update $\boldsymbol{V}_{1_{temp}}^{t+1}$ by $\boldsymbol{V}_{1_{temp}}^{t+1} = \boldsymbol{V}_{1_{temp}}^t + \alpha \frac{\partial L}{\partial \boldsymbol{V}_{1_{temp}}^t}$.
    Check the convergence conditions: $\|\boldsymbol{V}_{1_{temp}}^{t+1} - \boldsymbol{V}_{1_{temp}}^t\|_\infty \leq \epsilon$.
    $t = t + 1$.
  $\boldsymbol{V}_1^{k+1} = \boldsymbol{V}_{1_{temp}}^t$
  Fix $\boldsymbol{V}_1^{k+1}$, update $\boldsymbol{V}_2^{k+1}$ by Eq. (32).
  Update $\boldsymbol{Y}_1^{k+1}$ by Eq. (33).
  Update $\boldsymbol{Y}_2^{k+1}$ by Eq. (34).
  $\mu^{k+1} = min(\rho\mu^k, \mu^{max})$
  Check the convergence conditions: $\|\boldsymbol{V}_1^{k+1} - \boldsymbol{V}_1^k\|_\infty \leq \varepsilon$, $\|\boldsymbol{V}_2^{k+1} - \boldsymbol{V}_2^k\|_\infty \leq \varepsilon$ $\|\boldsymbol{Y}_1^{k+1} - \boldsymbol{Y}_1^k\|_\infty \leq \varepsilon$,$\|\boldsymbol{Y}_2^{k+1} - \boldsymbol{Y}_2^k\|_\infty \leq \varepsilon$
  $k = k + 1$.
  $t = 0$
return $\boldsymbol{V}_1^k$
Perform spectral clustering on $\boldsymbol{V}\boldsymbol{V}^T$ to obtain $\boldsymbol{Y}_{pred}$

standard pairwise affinity is unsatisfactory when the feature dimension increases. In comparison, the proposed UTC can perform clustering tasks stably by integrating affinities of different orders. Finally, we conduct experiments on six real datasets to verify the effectiveness of the proposed UTC by comparison with five popular clustering methods.

4.1 Experimental Settings

We employ five popular clustering methods to benchmark the proposed UTC. A brief introduction to these methods is given below:

- **Spectral clustering (SC)** [3]: The classic spectral clustering gives a baseline on behalf of pairwise similarity.
- **Clustering with Adaptive Neighbors (CAN)** [17]: Dynamically learning the affinity matrix by assigning the adaptive neighbors for each data point based on local distance. The final similarity matrix's connected components are made precisely equal to the cluster number by imposing a new rank constraint.
- **Clique averaging+ncut (CAVE)** [21]: The hypergraph, on behalf of the high-order affinity, is approximated using clique averaging, and the resulting graph is partitioned using the normalized cuts algorithm.
- **Pair-to-pair clustering (PPC)** [26]: Tetradic undecomposable affinities are used to obtain the low-dimension embedding as the final similarity matrix of the spectral method.
- **Integrating tensor similarity and pairwise similarity (IPS2)** [26]: Applying the spectral clustering method on the fused similarity that combines the pair-to-pair affinities and pairwise similarity.
- **Uniform Tensor Clustering (UTC)**: The proposed method.

In terms of the type of the affinity, the five methods can be divided into three categories, i.e., pairwise affinity, high-order affinity, and fused-order affinity.

The performance of each method is evaluated with respect to five popular metrics: accuracy (ACC), adjusted Rand index (ARI), F-score, normalized mutual information (NMI), and purity (PURITY). The larger the value is, the better the performance is.

4.2 Experiment on Synthdata-1 to Validate the Discrimination Potential of UTC via Triadic Affinity

Syndata-1 consists of 40 samples from two different subgroups. Each group is perfectly distributed along a straight line, while the two groups are distributed perpendicularly to each other. We use circles and crosses to denote the samples from the two subgroups, and their distribution is shown in Fig. 3(a).

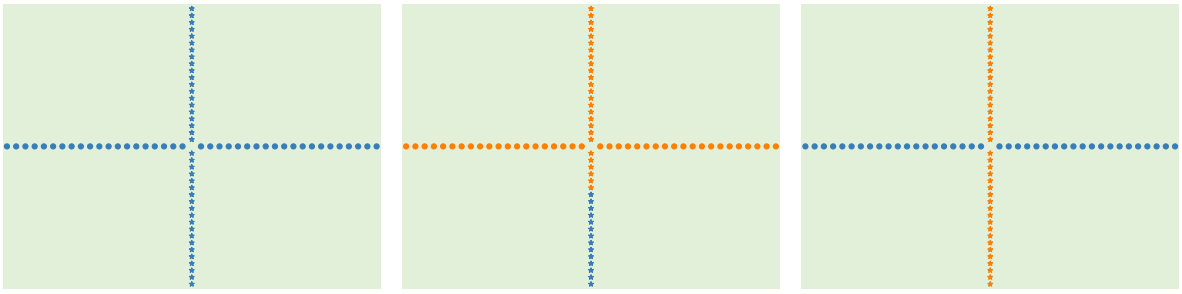
We applied the spectral clustering, which works on pairwise affinity, on **Syndata-1**. Since the samples from the two subgroups are perpendicular, the resulting affinity matrix is non-differentiable. Therefore, the SC fails to yield the correct assignment. It mistakenly classifies the samples into two parts, with most in the upper and the rest in the bottom. In comparison, UTC can identify the sample assignments perfectly with 100% accuracy. The results indicate that the proposed undecomposable high-order affinity can explore complementary information to improve the clustering performance.

4.3 Experiment on Synthdata-2 to Validate the Stability of UTC over Dimension Expanding via High-order Affinities

Syndata-2 consists of three subgroups, each derived from independent and identically distributed normal distributions with an equal standard deviation of 0.5 and mean of 2. The data from different groups are orthogonal, and each group contains 30-40 samples.

The pairwise affinity tends to be ambiguous with expanding feature dimensionality, limiting the clustering performance. We applied the proposed UTC on **Syndata-2** to validate its stability in handling high-dimensional data by comparison with popular methods, including SC, CAN and CAVE, PPS, and IPS2. The results are shown in Fig. 4. When the feature dimension varies from 10 to 10000, UTC can maintain robustness to perform accurate clustering. Furthermore, the baseline methods gradually lose the ability to identify the data structure.

Moreover, when the number of data dimensions reaches 10000, Tab. 2 shows that our proposed UTC methods, including fusing pairwise with triadic affinity, tetradic affinity, and both, substantially outperform all other state of the art methods. The ACC clustering results of fusing triadic and fusing tetradic are higher than those of the second-highest baseline method by 14% and 28%, demonstrating the effectiveness of fusing high-order affinity with pairwise relation under HDLSS data. Additionally, the ACC



(a) (b) (c)

Fig. 3: Experiment on Syndata-1. (a) Syndata-1 is constructed to consist of two subgroups, which are perfectly distributed along two cross-lines, highlighted by $\cdot$ and $*$; (b) The assignments obtained using pairwise affinity mistakenly mix the two groups, highlighted in orange and blue. Most of the samples are labeled incorrectly; (c) The assignments obtained by UTC using triadic affinity perfectly segment the samples into the correct subgroups, highlighted in orange and blue.

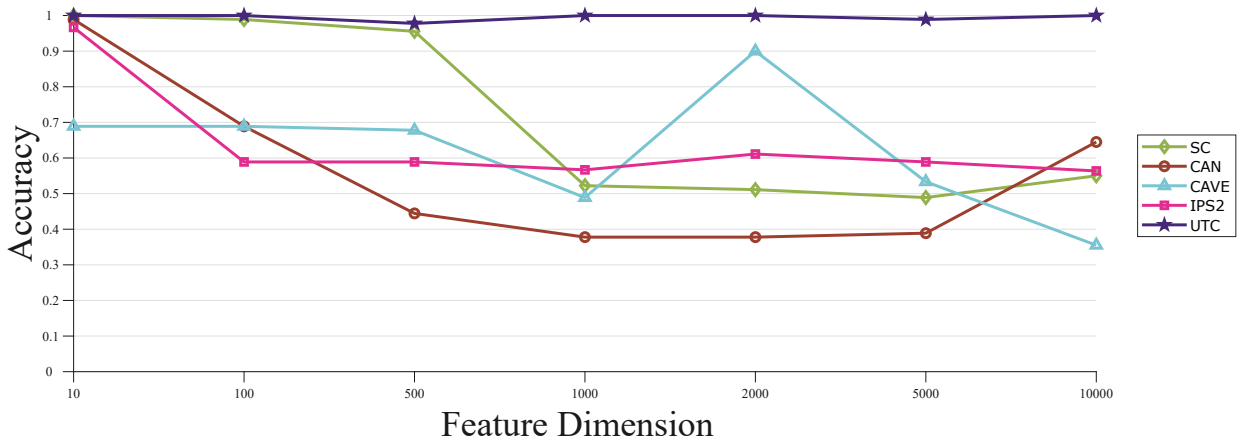


Fig. 4: Experiment to demonstrate the stability of UTC when dealing with high-dimensional feature corruption. The feature dimension is artificially increased from 10 to 10000. The accuracy of most of the comparative methods gradually decreased from 100% to around 35%. In comparison, the performance of UTC was stable; the accuracy was maintained at approximately 100% as the dimension increased.

TABLE 2: Clustering results on Syndata-2 when the dimensionality is 10000

Datasets	Methods	ACC	ARI	F-SCORE	NMI	PURITY
Syndata2	SC	0.55	0.4427	0.6633	0.6544	0.68
	CAN	0.6449	0.2306	0.5342	0.3323	0.6449
	CAVE	0.3551	-0.013	0.3734	0.0018	0.3551
	PPS	0.5636	0.3166	0.4472	0.4453	0.5636
	IPS2	0.7218	0.4847	0.5848	0.5862	0.7818
	UTC(Fusing pairwise and triadic affinity)	0.86	0.6719	0.7865	0.759	0.86
	UTC(Fusing pairwise and tetradic affinity)	0.77	0.5204	0.6799	0.5776	0.77
	UTC	**1**	**1**	**1**	**1**	**1**

achieves 100% when utilizing both higher-order affinities, proving that fusing different-order affinities is more helpful in completely explaining the data structure than is merging only a single higher-order affinity.

We also employ the t-SNE to visualize the differences in the embedded features after different methods. Most of the samples mix together when visualizing their embedding by t-SNE on the raw data, as shown in Fig. 5(a). On the other hand, the pairwise affinity, shown in Fig. 5(b), has no clear-cut structure, except the diagonal elements having large values. Therefore, it is challenging to discriminate the three subgroups when using the pairwise affinity matrix directly. Now, we fuse the three-order tensor affinity with pairwise affinity and apply UTC to obtain a local embedding. t-SNE is applied to the resulting embedding to visualize the sample assignment distribution. The visualized result is shown in Fig. 5(c). The samples in one subgroup (Subgroup A) are clearly separated from the others, while the remaining two subgroups (Subgroups B and C) are not well-distinguished. The upper left of the affinity matrix shows an obvious blocky structure, but the remaining part lacks a clear structure, as shown in Fig. 5(d), which confirms the results obtained by t-SNE. Alternatively, if we fuse the fourth-order tensor affinity with the pairwise matrix

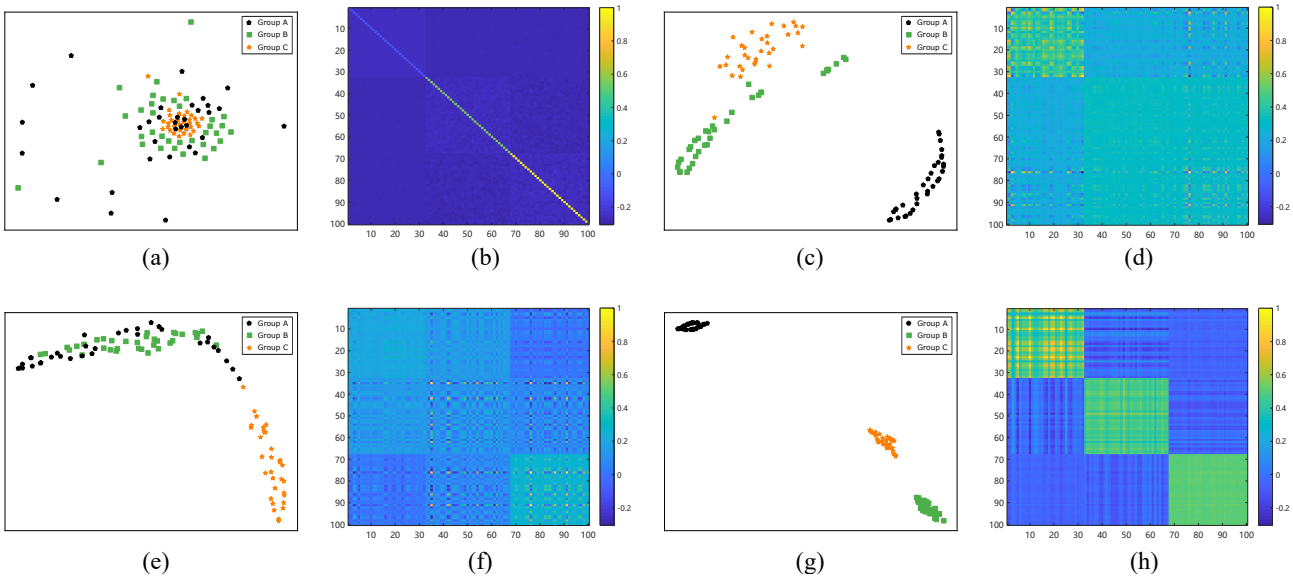


Fig. 5: Visualization results on Syndata2 with a feature dimensionality of 10000. We apply t-SNE on the embedding features derived from UTC and the corresponding affinity matrix in three ways. Clearly, most of the samples are mixed together when applying t-SNE to the raw data(a), and the corresponding affinity matrix has no clear-cut structure (b). The samples in group A are well-segmented, while the other groups remain undistinguished when fusing the triadic affinity with pairwise affinity (c). The upper left of the corresponding affinity matrix has blocky structures (d). Although group C can be identified accurately by combining fourth-order association with pairwise similarity, the remaining two types of the Gaussian mixture distribution are merged (e),(f). By combining the third-order and fourth-order affinities with pairwise similarity, each group can be identified accurately (h), and three distinct blocks are presented on the diagonal of the corresponding affinity matrix (g).

TABLE 3: Statistics on six tested real-world datasets.

Dataset	Type	Samples	Features	Clusters
Yale	Facial Image	55	4096	5
Coil20	Object Image	100	16384	5
Lymphoma	Bioinformatics	62	4702	3
DriveFace	Facial Image	78	307200	3
Lung	Bioinformatics	100	12600	3
GLI-85	Bioinformatics	85	22283	2

affinity and apply UTC to obtain another local embedding, the visualization using t-SNE is shown in Fig. 5(e). The samples in Subgroup C clearly are separated from the other two subgroups, while the other two subgroups are merged. The affinity heatmap also validates the observation that Subgroup A in the lower-right corner possesses distinct value compared to the others, as shown in Fig. 5(f). Finally, we fuse the pairwise affinity with the third-order and fourth-order. UTC is employed to learn a uniform embedding that is then visualized via t-SNE, as shown in Fig. 5(g). The three subgroups are perfectly separated in this case. The diagonal elements of the corresponding affinity matrix also show three obvious block-like structures and an affinity difference between the three subgroups. Therefore, an accurate sample assignment can easily be obtained from the embedding by the proposed UTC. In summary, the above experimental results validate that the high-order tensor affinities make up for the insufficiency of the pairwise affinity matrix when corrupted by high-dimensional features.

4.4 Experiments on Real Datasets

We then validated the power of UTC by applying it to six public benchmark datasets. The six datasets are chosen to be representative of various sources, including facial image, object image, and bioinformatics data. The statistics for the six datasets are summarized in Tab. 3. Additionally, a detailed description is provided.

- **Yale** dataset has 165 grayscale images covering 15 different individuals. Each individual has 11 different facial and configuration images, such as with glasses, without glasses, center-light, left-light, happy, sad, and so on. We extracted 4096-dimensional raw pixel values of every image in 5 classes for our experiment.
- **Coil20** dataset contains 1440 images of 20 categories of objects. Each category has 72 images from different views. We sample 20 samples from 5 classes, and all images have a size of 128×128.
- **Lymphoma** dataset, one of the most common subtypes of non-Hodgkin's lymphoma has two molecularly different forms of diffuse large B-cell lymphoma (DLBCL), which have gene expression patterns that indicate different stages of B cell differentiation. The dataset contains a total of 62 samples, 4702 based on expression fragments.
- **DriveFace** dataset contains images sequences of subjects while driving in real scenarios. It is composed of 606 samples of 6400 features each, acquired over different days from 3 drivers with various facial features, such as glasses and beards. For each type, we pick 26 samples for the experiment.
- **Lung** dataset contains a total of 203 samples that can be divided into 5 classes, with 149, 21, 20, 6, and 17 samples, respectively. Each sample has 12,600 genes. We selected 40 samples of the first category and all samples of the remaining four categories for experiments.

TABLE 4: Clustering performance on six real datasets

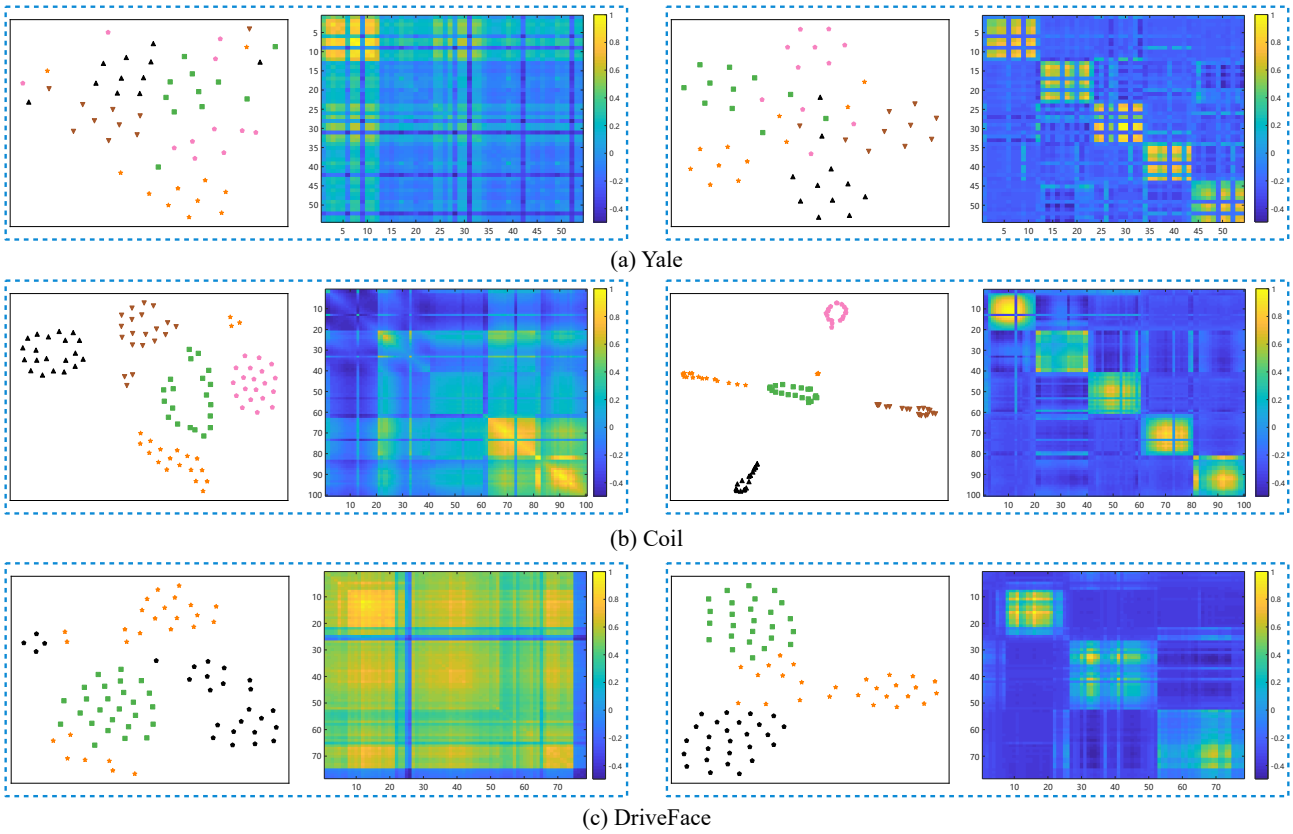
Datasets	Methods	ACC	ARI	F-SCORE	NMI	PURITY
YALE	SC	0.4727	0.2528	0.423	0.4097	0.4909
	CAN	0.5091	0.3316	0.4839	0.5046	0.5273
	CAVE	0.5818	0.2871	0.4337	0.4307	0.6182
	PPS	0.5636	0.3166	0.4472	0.4453	0.5636
	IPS2	0.7218	0.4847	0.5848	0.5862	0.7818
	UTC	**0.8**	**0.5019**	**0.6036**	**0.67**	**0.8**
Coil	SC	0.83	0.9068	0.9248	0.9352	0.96
	CAN	0.86	0.7660	0.7807	0.8886	0.81
	CAVE	0.84	0.6852	0.7462	0.7575	0.84
	PPS	0.93	0.8383	0.8694	0.869	0.93
	IPS2	**0.97**	**0.9280**	**0.9419**	**0.9460**	**0.97**
	UTC	**0.97**	**0.9280**	**0.9419**	**0.9460**	**0.97**
DriveFace	SC	0.641	0.3304	0.5921	0.4505	0.641
	CAN	0.7436	0.3671	0.5941	0.4486	0.7436
	CAVE	0.8974	0.7274	0.8182	0.7449	0.8974
	PPS	0.7436	0.4655	0.6433	0.5144	0.718
	IPS2	0.7436	0.4711	0.6549	0.5731	0.7436
	UTC	**0.9615**	**0.8869**	**0.9237**	**0.8675**	**0.9615**
Lymphoma	SC	0.8065	0.5059	0.7167	0.6634	0.8548
	CAN	0.9839	0.9471	0.9733	0.9255	0.9839
	CAVE	0.8065	0.5060	0.7167	0.6634	0.8548
	PPS	0.9194	0.7570	0.8696	0.7823	0.9194
	IPS2	0.9839	0.9471	0.9733	0.9255	0.9839
	UTC	**1**	**1**	**1**	**1**	**1**
Lung	SC	0.8	0.5855	0.675	0.6848	0.81
	CAN	0.77	0.5973	0.6971	0.6995	0.85
	CAVE	0.8	0.5855	0.675	0.6848	0.81
	PPS	0.73	0.541	0.6472	0.6311	0.8
	IPS2	0.82	0.541	0.7427	**0.7289**	0.85
	UTC	**0.85**	**0.6941**	**0.7634**	0.7219	**0.85**
GLI-85	SC	0.6471	0.0731	0.5721	0.1385	0.6941
	CAN	0.6588	-0.0353	0.697	0.042	0.6941
	CAVE	0.7412	0.1225	0.7376	0.171	0.7412
	PPS	0.5294	-0.0447	0.5777	0.1103	0.6941
	IPS2	0.7294	0.2012	0.6267	0.2754	0.7294
	UTC	**0.8118**	**0.3772**	**0.7177**	**0.2798**	**0.8118**

- **GLI-85** dataset consists of the transcriptional data of 85 diffuse infiltrating gliomas from 74 patients. Those gliomas can be divided into two kinds of tumor subclasses. Each instance has 22283 features.

We applied the proposed UTC as well as five comparative methods on the six datasets, and the results are summarized in Tab. 4. The bold value represents the best result. The clustering method that uses higher-order affinity uniformly outperforms the standard method using pairwise affinity. Therefore, the higher-order affinity constructed from the relationships of multiple samples can better describe the spatial structure of the data, which is advantageous when dealing with HDLSS data. Furthermore, our proposed UTC method significantly outperforms the competitive methods with respect to all five evaluation metrics for all types of data, including facial image, object image, and bioinformatics data.

On the Yale dataset, the ACC clustering result of our method is over 33% higher than that of SC and over 8% higher than the second-highest IPS2. On the GLI-85 dataset, our results are more than 16% and 6% higher than those of the baseline and second-highest IPS2 methods. There are also 1%, 7%, 10%, 2%, and 3% improvements compared with the second-best performance on the Coil, DriveFace, Colon, Lymphoma, and Lung datasets, respectively. IPS2 and UTC achieve the top two results on almost all datasets with respect to nearly all the metrics.

We also visualized the sample spatial distribution after t-SNE[31] and the heatmap of the affinity matrix on the Yale, Coil, and DriveFace datasets in Fig. 6 and the Lymphoma, Lung, and GLI-85 datasets in Fig. 7, respectively. The spatial distributions after t-SNE on the raw data and the resulting embedding are shown on the left side of the two columns, while the heatmaps of the raw data and the affinity matrix fusing with various orders are on the other side. In most figures, the low-dimensional embedding after UTC displays better separation and less overlap, thus making it easy for clustering. In comparison, the heatmap of similarity produced by SC and UTC further demonstrates the effectiveness of our method. Deep yellow represents a large proximity value. The affinity heatmap on the raw samples possesses highly blurred boundaries, and there is no apparent block structure. However, the affinity matrix calculated from the low-dimensional embedding after UTC has clear-cut boundaries. The large value in the matrix generated by UTC is concentrated on the diagonal blocks, implying that the fused affinity can more accurately reflect the real relationship among the samples, thus yielding superior clustering performance in Tab. 4. Taking the Yale dataset as an example, the ACC of the clustering results obtained by UTC is 33% higher than that of SC. The visualization results confirm our method's superiority intuitively, as shown in Fig. 6(a). Most of the samples are mixed together in the visualization based on pairwise affinity, resulting in an inability to distinguish between different subgroups accurately, as shown on the far left of Fig. 6(a). The heatmap of pairwise affinity confirms that only one cluster is significantly different from the other samples. Correspondingly, in the t-SNE visualization of the resulting embedding, most samples are clustered with their same cluster's partners. Five obvious blocks are displayed on the diagonal in the heatmap. The right



(a) Yale

(b) Coil

(c) DriveFace

Fig. 6: Visualization of the latent representations with t-SNE and corresponding affinity heatmaps of the (a) Yale; (b) Coil; and (c) DriveFace datasets. The first and second columns are the visualizations of the embedding by t-SNE on the raw data and the samples' affinity heatmap. The third and the fourth columns are the visualization results of the obtained uniform embedding by UTC and the samples' affinity heatmap, respectively.

side of Fig. 6(a) confirms that UTC can better reflect the real structure of the data compared with single-order methods, which is critical for clustering.

In summary, the reason for the superior performance of UTC is that it utilizes the complementarity of high- and low-order affinities to comprehensively capture the spatial structure of data, thereby leading to a decisive effect on the clustering performance.

4.5 Computational Complexity Analysis

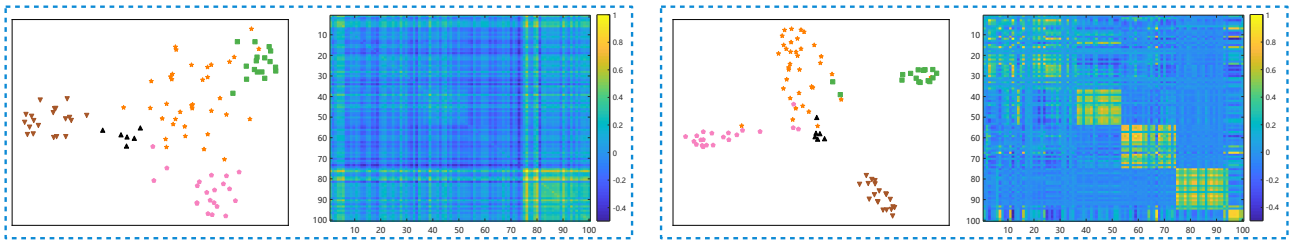
The UTC consists of two major computational components: constructing the high-order tensor affinity and solving the minimization problem in Eq. (27). In the first component, we calculate both the triadic and tetradic affinities by means of their k-nearest neighbors (KNN). Given m samples, one can use the KNN strategy to reduce the time complexity from $\mathcal{O}(m^4)$ to $\mathcal{O}(mk^4)$. Thus, the two affinity tensors are sparse, with the number of nonzero elements being s. For the latter, we solve th model by means of an iterative strategy. The subproblem of $\boldsymbol{V}_1$ based on the gradient descent strategy has a computational complexity of $\mathcal{O}(ncsm^2)$, where the c denotes the number of clusters and the maximum iteration number is n. Solving the other sub-problem $\boldsymbol{V}_2$ takes $\mathcal{O}(csm^2)$ using the Germs algorithm. Thus, each iteration has a total computational cost of $\mathcal{O}(ncsm^2) + \mathcal{O}(csm^2) = \mathcal{O}((n+1)ncsm^2)$. In total, the complexity to learn the uniform embedding in Algorithm 1 is $\mathcal{O}(mk^4 + K((n+1)ncsm^2))$, with K being the number of iterations.

5 Conclusion

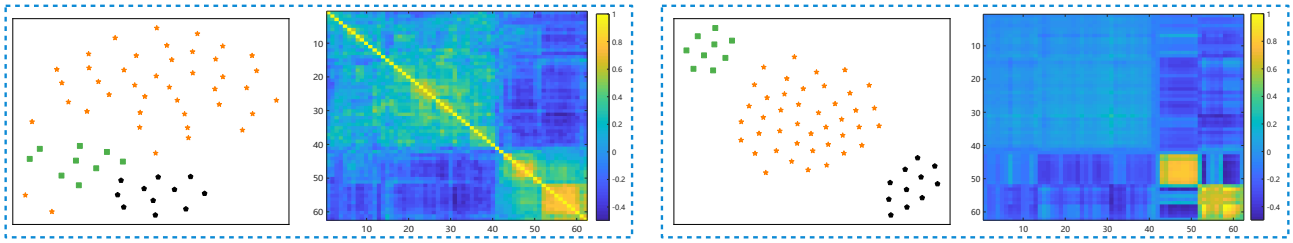
The clustering of small-size samples with large dimensions is a bottleneck problem. We propose a unified learning model to effectively fuse multiple samples' affinities of different orders to obtain the sample's uniform low-dimensional embedding. Tensor-vector products uniformly formulate the sample affinities of various orders, which opens a new door for studying multiple samples. Our method involves three matrix products: arithmetical, Khatri-Rao and Kronecker product. They are jointly optimized to yield a uniform low-dimensional embedding of the samples. Experiments on synthetic data demonstrate the power of fusing different-order affinities and experiments on several real datasets with small sample size yet large feature dimensionality show the effectiveness and superiority of the method over other popular approaches.

Acknowledgment

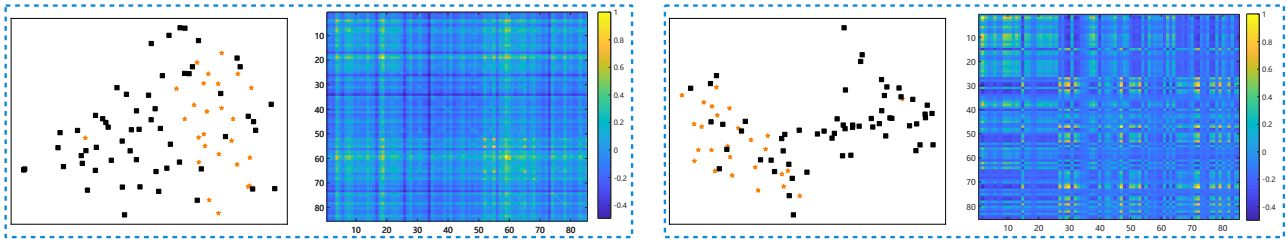
This work was supported in part the National Natural Science Foundation of China (62102153, 62172112, U21A20520), by the National Key Research and Development Program of China under Grant 2019YFB2102102, the Fundamental Research Fund for the Central Universities (x2jsD2200720), and the Key-Area Research and Development of Guangdong Province under Grant 2020B1111190001.



(a) Lung



(b) Lymphoma



(c) GLI-85

Fig. 7: Visualization of the latent representations with t-SNE and corresponding affinity heatmaps of the (a) Lung; (b) Lymphoma, and (c) GLI-85 datasets. The first and second columns are the visualization by t-SNE of the raw data and the samples' affinity heatmap. The third and fourth columns are the visualization results of the obtained uniform embedding by UTC and the samples' affinity heatmap, respectively.

REFERENCES

[1] T. Hofmann and J. M. Buhmann, "Pairwise data clustering by deterministic annealing," *Ieee transactions on pattern analysis and machine intelligence*, vol. 19, no. 1, pp. 1–14, 1997. (document)

[2] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 881–892, 2002. (document)

[3] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000. (document), 2.2, 4.1

[4] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in neural information processing systems*, 2002, pp. 849–856. (document)

[5] H. Li, S. Tian, Y. Li, Q. Fang, R. Tan, Y. Pan, C. Huang, Y. Xu, and X. Gao, "Modern deep learning in bioinformatics," *Journal of molecular cell biology*, vol. 12, no. 11, pp. 823–827, 2020. (document)

[6] T. Tian, J. Wan, Q. Song, and Z. Wei, "Clustering single-cell rna-seq data with a model-based deep learning approach," *Nature Machine Intelligence*, vol. 1, no. 4, pp. 191–198, 2019. (document)

[7] X. Peng, J. Feng, J. T. Zhou, Y. Lei, and S. Yan, "Deep subspace clustering," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2020. (document)

[8] Y. Lu, Y. Cheung, and Y. Y. Tang, "Self-adaptive multiprototype-based competitive learning approach: A k-means-type algorithm for imbalanced data clustering," *IEEE Transactions on Cybernetics*, pp. 1–15, 2019. (document)

[9] W. Yang, C. Hui, D. Sun, X. Sun, and Q. Liao, "Clustering through probability distribution analysis along eigenpaths," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018. (document)

[10] Y. Chen, L. Zhou, S. Pei, Z. Yu, Y. Chen, X. Liu, J. Du, and N. Xiong, "Knn-block dbscan: Fast clustering for large-scale data," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2019. (document)

[11] X. Xu, J. Li, M. Zhou, J. Xu, and J. Cao, "Accelerated two-stage particle swarm optimization for clustering not-well-separated data," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 11, pp. 4212–4223, 2018. (document)

[12] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *International conference on database theory*. Springer, 2001, pp. 420–434. (document)

[13] D. François, V. Wertz, and M. Verleysen, "The concentration of fractional distances," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 7, pp. 873–886, 2007. (document)

[14] S. Sarkar and A. K. Ghosh, "On perfect clustering of high dimension, low sample size data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. (document)

[15] P. Borysov, J. Hannig, and J. S. Marron, "Asymptotics of hierarchical clustering for growing dimension," *Journal of Multivariate Analysis*, vol. 124, pp. 465–479, 2014. (document)

[16] P. Hall, J. S. Marron, and A. Neeman, "Geometric representation of high dimension, low sample size data," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 3, pp. 427–444, 2005. (document)

[17] F. Nie, X. Wang, and H. Huang, "Clustering and projected clustering with adaptive neighbors," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 977–986. (document), 4.1

[18] H. Wang, G. Xiao, Y. Yan, and D. Suter, "Searching for representative modes on hypergraphs for robust geometric model fitting," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 3, pp. 697–711, 2018. (document)

[19] H. C. Nguyen and H. Mamitsuka, "Learning on hypergraphs with sparsity," *IEEE transactions on pattern analysis and machine intelligence*, 2020. (document)

[20] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: Clus-

tering, classification, and embedding," *Advances in neural information processing systems*, vol. 19, pp. 1601–1608, 2006. (document)

[21] S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D. Kriegman, and S. Belongie, "Beyond pairwise clustering," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. IEEE, 2005, pp. 838–845. (document), 4.1

[22] S. R. Bulo and M. Pelillo, "A game-theoretic approach to hypergraph clustering." in *NIPS*, vol. 22. Citeseer, 2009, pp. 1571–1579. (document)

[23] Z. Zhang, H. Lin, Y. Gao, and K. BNRist, "Dynamic hypergraph structure learning." in *IJCAI*, 2018, pp. 3162–3169. (document)

[24] G. Li, L. Qi, and G. Yu, "The z-eigenvalues of a symmetric tensor and its application to spectral hypergraph theory," *Numerical Linear Algebra with Applications*, vol. 20, no. 6, pp. 1001–1029, 2013. (document)

[25] D. Ghoshdastidar and A. Dukkipati, "Spectral clustering using multilinear svd: Analysis, approximations and applications," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015. (document)

[26] H. Peng, Y. Hu, J. Chen, W. Haiyan, Y. Li, and H. Cai, "Integrating tensor similarity to enhance clustering performance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. (document), 2.3, 2.3, 2.1, 3, 4.1

[27] S. Rabanser, O. Shchur, and S. Günnemann, "Introduction to tensor decompositions and their applications in machine learning," *arXiv preprint arXiv:1711.10781*, 2017. 2.1

[28] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007. 2.2

[29] S. Sarkar and A. K. Ghosh, "On perfect clustering of high dimension, low sample size data," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 9, pp. 2257–2272, 2019. 2.3

[30] A. Kumar, P. Rai, and H. Daume, "Co-regularized multi-view spectral clustering," *Advances in neural information processing systems*, vol. 24, pp. 1413–1421, 2011. 3.1.2

[31] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008. 4.4