ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

HDBSCAN for 3rd-order tensors

Dina Faneva Andriantsiory[1], Joseph Ben Geloun[1] and Mustapha Lebbah[2]

1- LIPN, UMR CNRS 7030 - Université Sorbonne Paris Nord
Villetaneuse - France

2- Paris-Saclay University, UVSQ, David Lab, Versailles - France

Abstract. Several methods for tensor clustering require hyperparameters such as the cluster size or the number of clusters per mode. These methods present a challenge because, for real datasets, such inputs cannot be determined without incurring significant costs. Recently, Multi-Slice Clustering (MSC) has addressed this issue by utilizing a threshold parameter to perform data clustering. MSC identifies signal slices that reside in a lower-dimensional subspace within a 3rd-order rank-1 tensor dataset. However, determining the tensor rank remains a complex task. The current work introduces a new approach to tensor clustering that can extract clusters of similar slices and is also capable of finding co-clustering and triclustering in 3rd-order tensors of any rank. Our algorithm is based on the density of the data.

1 Introduction

From an algebraic point of view, an n-way or n-th order tensor is an element of the tensor product of n vector spaces, each of which has its own coordinate system [1]. The tensor order indicates the number of dimensions in the array. As a data structure, consider n_1 individuals with n_2 features and collect the data for each individual-feature pair at n_3 different times. This is an example of a dataset structured in three dimensions. A convenient way to encode such data is given by a 3rd-order tensor. Multidimensional data of this nature arises in several contexts such as neuroscience [2] and computer vision [3]. To analyze this data without having a detailed understanding of it, we use unsupervised learning. Clustering is one of the most popular unsupervised learning methods for extracting relevant information as the structural similarity in the dataset. It operates by segmenting the dataset into significant groups or clusters. Thus, a variety of computational methods have been developed for clustering multidimensional data, ranging from matrices to higher-order tensors, as documented in the literature [1, 4, 5, 6, 7].

The general structure of clustering algorithms requires the data to be treated but also the hyperparameters in the inputs: cluster sizes or the number of the clusters, as exemplified by the tensor biclustering [5] and the Multiway clustering via tensor block models (TBM) [8]. These hyperparameters are not easy to set, and more to the point, especially difficult to set for real data. Moreover, their values influence the quality of the algorithm's output. In [9], the authors provide a method that replaces the number of clusters with a measure of similarity within a cluster. This method is called Multi-Slice Clustering (MSC) and it performs on 3rd-order tensors. There is however a limitation of the application domain of the MSC method. Indeed, the latter is designed to find one cluster within rank-1 tensor datasets. Such a condition is equally difficult to verify for real data [1]. Therefore, it is essential to have an efficient clustering method with the higher-rank tensor dataset.

In this work, we introduce a novel clustering algorithm tailored for 3rd-order tensor datasets. Our method leverages the density of the data to discern multiple clusters within a tensor dataset, irrespective of its rank. We have designated this technique as HDBSCAN for tensors (HDBSCAN-Tensor). As implied by its designation, our approach entails representing the tensor as a matrix, where each row corresponds to one slice from the original tensor. Subsequently, we employ the HDBSCAN algorithm, Hierarchical Density-Based Spatial Clustering of Applications with Noise [10], to pinpoint groups of slices exhibiting high similarity (refer to figure 1). The process commences with the eigendecomposition of each slice, selecting the most substantial eigenvalue and its associated eigenvector to represent the slice within the matrix. The HDBSCAN algorithm then utilizes this matrix as the input data. Our algorithm necessitates the tensor data and an integer parameter for HDBSCAN's operation. Upon comparing our method's performance with other tensor clustering techniques, we have ascertained that it yields competitive results.

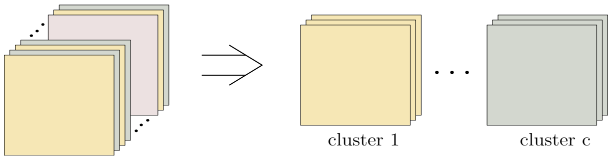

Fig. 1: Clustering of the slices of a 3-rd tensor dataset

This paper is structured as follows: In section 2, we enumerate our notation and elaborate on our model. In section 3, we show the experimental results of our algorithm and benchmark the quality of these results against other methods. Finally, in section 4, we summarize this work and offer future perspectives.

2 Notation and model

2.1 Notation

In this paper, we use $\mathcal{T}, \mathcal{X}, \mathcal{Z}$ to denote the dataset's 3rd-order tensor. The Matlab notation is employed for the tensor entries. Matrices are denoted by the uppercase letters M, T, and the Frobenius norm of a matrix M is represented by $\|M\|_F$. The boldface lowercase letters $\mathbf{u}, \mathbf{v}, \mathbf{w}, \ldots$ signify vectors, while the lowercase letters $n, \gamma, \lambda, \sigma$ signify scalars. For a positive integer n, we define $[n] = \{1, \ldots, n\}$.

2.2 Model

Our primary focus will be on 3rd-order tensors. Thus, let $\mathcal{T} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ be our tensor dataset. We decompose it as $\mathcal{T} = \mathcal{X} + \mathcal{Z}$, where $\mathcal{X}$ is called the signal tensor and $\mathcal{Z}$ the noise tensor. According to the CANDECOMP/PARAFAC (CP) decomposition of a tensor [1], if the signal tensor is a sum of r rank-1 tensors, then $\mathcal{T}$ can be written as the following.

$$\mathcal{T} = \mathcal{X} + \mathcal{Z} = \sum_{i=1}^{r} \gamma_i \, \mathbf{w}_i \otimes \mathbf{u}_i \otimes \mathbf{v}_i + \mathcal{Z} \tag{1}$$

where $\forall i, \gamma_i > 0$ stands for the signal weight, $\mathbf{w}_i \in \mathbb{R}^{n_1}, \mathbf{u}_i \in \mathbb{R}^{n_2}$ and $\mathbf{v}_i \in \mathbb{R}^{n_3}$ are unit vectors. $\mathbf{w}_i$ and $\mathbf{w}_j$ are orthogonal for all $i \neq j$, the same property is verified for the vector $(\mathbf{u}_i)_i$ and the vector $(\mathbf{v}_i)_i$, the entries of $\mathcal{Z}$ are independent identically distributed (i.i.d) standard Gaussian random variable.

The mode-1 slices of $\mathcal{T}$ form a matrix in $\mathbb{R}^{n_2 \times n_3}$, expressed as $\mathcal{T}(i,:,:) = \mathcal{X}(i,:,:) + \mathcal{Z}(i,:,:)$ for all $i \in [n_1]$. In the following, we denote by T_i, X_i and Z_i the i-th mode-1 slice of the tensor $\mathcal{T}, \mathcal{X}$ and $\mathcal{Z}$, respectively.

The MSC approach emphasizes the representation of each slice by its largest eigenvalue and the corresponding eigenvector (also called largest eigenvector) for a rank-one tensor. Lemma 6 in [5] demonstrates that the infinity norm of the difference between the largest eigenvector of the covariance matrix of T_i and X_i approaches zero as the value of n_3 increases sufficiently. Furthermore, the largest eigenvector of $X_i^t X_i$ is shown to be equal to $\mathbf{v}_i$.

Let us assume that our signal tensor is the sum of two rank-one tensors:

$$\mathcal{X} = \gamma_1 \mathbf{w_1} \otimes \mathbf{u}_1 \otimes \mathbf{v}_1 + \gamma_2 \mathbf{w_2} \otimes \mathbf{u}_2 \otimes \mathbf{v}_2, \tag{2}$$

and $X_i = \gamma_{1i} \mathbf{u}_1 \mathbf{v}_1^t + \gamma_{2i} \mathbf{u}_2 \mathbf{v}_2^t$, with $\gamma_{ji} = \gamma_j \mathbf{w}_j(i)$ for $j \in [2]$. This implies that the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ belongs to the eigenspace of the covariance matrix of X_i corresponding to the eigenvalues γ_{1i}^2 and γ_{2i}^2, respectively. However, we cannot assert that $\mathbf{v}_1$ is the largest eigenvector of $X_i^t X_i$. Indeed, referencing the definition in equation (1), we derive:

$$\begin{aligned} X_i^t X_i &= (\gamma_{1i} \mathbf{u}_1 \mathbf{v}_1^t + \gamma_{2i} \mathbf{u}_2 \mathbf{v}_2^t)^t (\gamma_{1i} \mathbf{u}_1 \mathbf{v}_1^t + \gamma_{2i} \mathbf{u}_2 \mathbf{v}_2^t) \\ &= \gamma_{1i}^2 \mathbf{v}_1 \mathbf{v}_1^t + \gamma_{2i}^2 \mathbf{v}_2 \mathbf{v}_2^t \end{aligned} \tag{3}$$

This reveals that the slicewise clustering approach may offer enhanced efficiency in determining subspace clustering for higher-rank tensor datasets.

The method that we propose is a combination of two methods: the construction of the eigenvector matrix, which is inspired by the MSC method [9], and the matrix clustering method HDBSCAN [10]. Each slice is represented by its largest eigenvalue and the corresponding eigenvector. For example, for the n_1 slices in mode-1 we have the following matrix,

$$M = [\tilde{\lambda}_1 \mathbf{v}_1 \quad \tilde{\lambda}_2 \mathbf{v}_2 \quad \cdots \tilde{\lambda}_{n_1} \mathbf{v}_{n_1}]^t \tag{4}$$

where we set $\tilde{\lambda}_i = \lambda_i/\lambda$, with λ_i is the largest eigenvalue of the covariance matrix $T_i^t T_i$ and $\lambda = \max_{i\in[n_1]} \lambda_i$.

We then apply the HDBSCAN method to the matrix M. This method is a clustering technique predicated on data density. It features a hyperparameter known as *min_cluster_size*, which denotes the minimum number of samples required in a group for that group to be considered a cluster. This parameter is specified at the input of our algorithm.

The HDBSCAN-Tensor method identifies the clusters of slices for each mode independently from other modes. Consequently, our algorithm delineates the clustering for a single mode (refer to algorithm 1). The same principle is applied to the remaining modes. Moreover, integrating the clustering results from two modes yields tensor coclustering, while amalgamating the results from all three modes delivers the multiway clustering of the tensor.

Algorithm 1 HDBSCAN for a 3rd-order Tensor

```
Data: The tensor data $\mathcal{T} \in \mathbb{R}^{n_1\times n_2\times n_3}$ and the value of the $min\_sample$
Result: The slice clustering $C = (C_1, \cdots C_k, \cdots)$
Inilialize the matrix $M$
$\lambda \leftarrow 0$
for $i = 1, \cdots, n_1$ do
    Center and reduce the columns of $T_i$
    Compute $Cov = T_i^t T_i$
    Compute the largest eigenvalue $\lambda_i$ and eigenvector $\mathbf{v}_i$ of $Cov$
    $M[i,:] \leftarrow \lambda_i \mathbf{v}_i$
    if $\lambda_i > \lambda$ then
        $\lambda \leftarrow \lambda_i$
    end
end
$M \leftarrow M/\lambda$
$C \leftarrow$ Compute the HDBSCAN algorithm to the matrix $M$
```

Complexity: We use the Rayleigh quotient iteration method to calculate the largest eigenvalue and its corresponding eigenvector, which incurs a computational complexity of $\mathcal{O}(n^2k)$ where k stands for the number of iteration [11]. The complexity of the HDBSCAN is $\mathcal{O}(n\log(n))$ [12]. Taking into account the 'for' loop, the overall complexity of our algorithm amounts to $\mathcal{O}(n^3k)$, with n being the number of data points.

3 Experiments

To assess the efficiency of our method, we run it on both synthetic and real datasets and benchmark the outcomes against those of the TBM method, the MSC method, and the Tucker+k-means method [7].

Synthetic data: Initially, we apply our algorithm to a synthetic dataset. The signal tensor is a sum of two rank-one tensors and it is generated according to the equation (1) such that for $i \in [2]$, $\mathbf{w}_i(j) = \mathbf{u}_i(j) = \mathbf{v}_i(j) = 1/\sqrt{s_i}$, for all $j \in [s_i]$, where s_i is the number of the nonzeros entries of $\mathbf{w}_i$, $\mathbf{u}_i$, and $\mathbf{v}_i$, respectively. The dimension of the tensor is fixed at $n_1 = n_2 = n_3 = 50$ and $s_i = 10$ for $i \in [2]$. The signal weight is varied from 20 to 75 in increments of 5. For each level of signal weight, the algorithm is executed 10 times. On each iteration, we calculate the Adjusted Rand Index to evaluate the clustering quality [13]. Subsequently, we compute the mean and standard deviation of the ten clustering quality assessments. The results are depicted in figure 2.

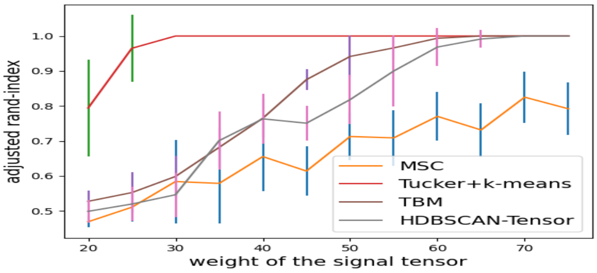

Fig. 2: The clustering quality for varying the signal weight.

We note that Tucker+k-means exhibits the best performance, which aligns with our expectations given that the data is constructed in line with the CP decomposition, which is a specific instance of the Tucker decomposition. The TBM method ranks slightly above HDBSCAN-Tensor. It is important to remember that these two methods incorporate the actual number of clusters as input. HDBSCAN-Tensor successfully identifies all clusters starting from $weight = 60$ by setting min_cluster_size=4. Lastly, MSC demonstrates the least effectiveness, which is anticipated since MSC is not designed for tensors of higher rank.

Real data: In this experiment, the real dataset concerns the experimental findings of Norgaard and Ridder (1994), who explored the challenge of measuring samples containing three distinct analytes using a flow injection analysis (FIA) system under a pH-gradient. The dataset forms a 12(samples)x100(wavelengths) x89(times) array. Our objective is to discern the clusters of slices along the time dimension. To appraise the outcomes, we calculate the root mean square error (RMSE) for each cluster. The findings are presented in table 1.

Table 1: The RMSE of the clusters

	mean of the RMSE	RMSE of the best cluster
HDBSCAN-Tensor	1.056	0.870
Tucker+k-means	1.403	0.906
TBM	1.152	1.060

Setting $min\_cluster\_size = 7$ for the HDBSCAN-Tensor method, it becomes obvious that our method outperforms the others. Note that to be able to compare the cluster quality, we fix the number of clusters in the other methods to be the number of clusters obtained in HDBSCAN-Tensor. As we vary the value of $min\_cluster\_size$, our method consistently maintains the best performance compared to the others. The code for our method is accessible at the provided repository link `https://depot.lipn.univ-paris13.fr/andriantsiory/hdbscantensor/-/tree/main`.

4 Conclusion

We developed a density-based tensor clustering algorithm for independently clustering single-mode slices, enabling co-clustering and multiway clustering when combining modes. Our experiments show its effectiveness and high performance compared to known clustering algorithms, particularly with real datasets of unknown ranks. Future work may explore scalability for larger datasets.

References

[1] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[2] Hua Zhou, Lexin Li, and Hongtu Zhu. Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association*, 108(502):540–552, 2013. PMID: 24791032.

[3] Yichuan Tang, Ruslan Salakhutdinov, and Geoffrey Hinton. Tensor analyzers. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 163–171, 2013.

[4] Victoria Hore, Ana Viñuela, Alfonso Buil, Julian Knight, Mark I McCarthy, Kerrin Small, and Jonathan Marchini. Tensor decomposition for multiple-tissue gene expression experiments. *Nature genetics*, 48(9):1094–1100, 2016.

[5] Soheil Feizi, Hamid Javadi, and David Tse. Tensor biclustering. In *Advances in Neural Information Processing Systems*, volume 30, pages 1311–1320, 2017.

[6] Timothy Carson, Dustin G. Mixon, and Soledad Villar. Manifold optimization for k-means clustering. In *2017 International Conference on Sampling Theory and Applications (SampTA)*, pages 73–77, 2017.

[7] Will Wei Sun and Lexin Li. Dynamic tensor clustering. *Journal of the American Statistical Association*, 114(528):1894–1907, 2019.

[8] Miaoyan Wang and Yuchen Zeng. Multiway clustering via tensor block models. *Advances in neural information processing systems*, 32, 2019.

[9] Dina Faneva Andriantsiory, Joseph Ben Geloun, and Mustapha Lebbah. Multi-slice clustering for 3-order tensor. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 173–178. IEEE, 2021.

[10] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*, pages 160–172. Springer Berlin Heidelberg, 2013.

[11] Lloyd N Trefethen and David Bau III. Numerical linear algebra. *Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA*, 1997.

[12] Leland McInnes and John Healy. Accelerated hierarchical density based clustering. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 33–42, 2017.

[13] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.