Fast Subspace Clustering Based on the Kronecker Product

Lei Zhou[1], Xiao Bai[1], Liang Zhang[1], Jun Zhou[2], Edwin Hancock[3]

1. School of Computer Science and Engineering and Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing, China
2. Information and Communication Technology, Griffith University, Nathan, Australia
3. Department of Computer Science, University of York, York, UK

ICPR 2020 — 25th International Conference on Pattern Recognition, Milan, Italy 10 | 15 January 2021

Abstract

Subspace clustering is a useful technique for many computer vision applications in which the intrinsic dimension of high-dimensional data is often smaller than the ambient dimension. Spectral clustering, as one of the main approaches to subspace clustering, often takes on a sparse representation or a low-rank representation to learn a block diagonal self-representation matrix for subspace generation. However, existing methods require solving a large scale convex optimization problem with a large set of data, with computational complexity reaches $O(N^3)$ for N data points. Therefore, the efficiency and scalability of traditional spectral clustering methods can not be guaranteed for large scale datasets. In this paper, we propose a subspace clustering model based on the Kronecker product. Due to the property that the Kronecker product of a block diagonal matrix with any other matrix is still a block diagonal matrix, we can efficiently learn the representation matrix which is formed by the Kronecker product of k smaller matrices. By doing so, our model significantly reduces the computational complexity to $O(kN^{3/k})$. Furthermore, our model is general in nature and can be adapted to different regularization based subspace clustering methods. Experimental results on two public datasets show that our model significantly improves the efficiency compared with several state-of-the-art methods. Moreover, we have conducted experiments on synthetic data to verify the scalability of our model for large scale datasets.

Method

1. Motivation

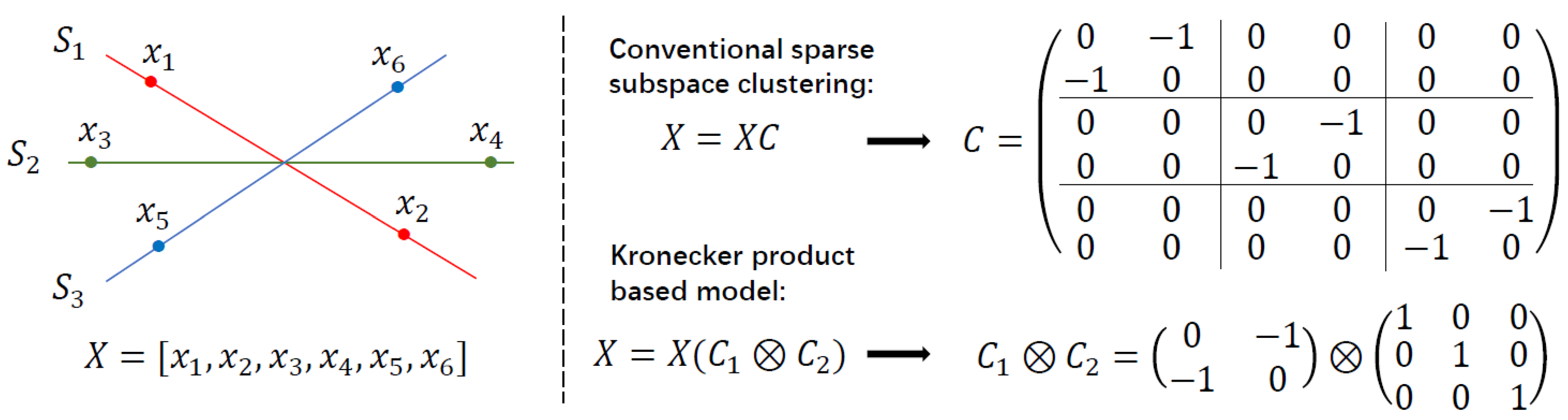

Fig.1 Left: Three 1D subspaces in $\mathbb{R}^2$ with normalized data points. Right: The solutions of conventional sparse subspace clustering method (upper) and our Kronecker product based model (lower). As shown, the space and computational complexity of our model achieve significant reduction compared with conventional method.

2. Formulation

We first introduce the Kronecker product. Let $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{p \times q}$, the Kronecker product of matrices A and B is $A \otimes B \in \mathbb{R}^{mp \times nq}$ which is defined as:

$$A \otimes B = \begin{bmatrix} a_{11} \times B & \cdots & a_{1n} \times B \\ \vdots & \ddots & \vdots \\ a_{m1} \times B & \cdots & a_{mn} \times B \end{bmatrix}$$

We assume that the self-representation matrix is formed by the Kronecker product of two smaller matrices C_1 and C_2. Here we use the important property that the Kronecker product of a block diagonal matrix with any other matrix is still a block diagonal matrix (as shown in Fig. 1). The optimization problem can be written as:

$$\min_{C_i} \|X - X(C_1 \otimes C_2)\|_F^2 + \lambda \|C_1 \otimes C_2\|_F^2$$

3. Optimization

We solve the optimization problem by updating each small matrix at a time, while keeping the other one fixed. Considering updating C_1, while C_2 is fixed, we start by rewriting

$$\begin{aligned} &\|X - X(C_1 \otimes C_2)\|_F^2 \\ =&tr((X - X(C_1 \otimes C_2))^T(X - X(C_1 \otimes C_2))) \\ =&\|X\|_F^2 - 2tr(X(C_1 \otimes C_2)X^T) \\ &+ tr(X(C_1 \otimes C_2)(X(C_1 \otimes C_2))^T) \end{aligned}$$

since $\|X\|_F^2$ is a constant, let

$$\Phi = -2tr(X(C_1 \otimes C_2)X^T) + tr(X(C_1 \otimes C_2)(X(C_1 \otimes C_2))^T)$$

then, the problem is equivalent to minimizing Φ. This can be transformed into a ridge regression problem which has optimal solution. We can solve C_2 in a similar manner to C_1, when C_1 is fixed. The computational complexity for this solution is $O(2N^{3/2})$.

When the number of small matrices is k, we can also solve it by updating one small matrix at a time, while keeping the remaining matrices fixed. In this situation, the problem is the same as k = 2 solved above. The computational complexity of the whole optimization is $O(kN^{3/k})$.

Experiments

We have conducted three sets of experiments on both real and synthetic datasets to verify the effectiveness of the proposed methods.

Table 1. The subspace clustering performance on the CMU PIE dataset.

No. Objects	5 Objects		10 Objects		20 Objects		40 Objects		60 Objects	
	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.
SSC	243.6	92.47	1182	89.25	3618	84.31	14502	82.37	-	-
KrSSC	**12.7**	91.28	**26.8**	88.27	**61.4**	83.86	**150.2**	81.75	**274.3**	79.48
LRR	216.4	94.53	852.5	92.14	2743	89.21	11463	85.47	-	-
KrLRR	**9.7**	92.51	**20.4**	90.72	**57.2**	88.13	**145.8**	85.21	**254.8**	83.65
TRR	152.7	97.35	548.2	96.05	2167	94.54	8427	91.74	-	-
KrTRR	**7.5**	95.21	**18.3**	94.52	**52.8**	93.84	**143.5**	90.23	**260.1**	87.26
NVR3	190.5	98.51	624.6	97.51	2536	95.75	11826	93.15	-	-
KrNVR3	**11.3**	97.14	**25.7**	96.26	**72.4**	93.96	**180.4**	91.57	**312.5**	89.15

Table 2. The subspace clustering performance on the MNIST dataset.

No. Points	500		1000		10000		30000		70000	
	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.
SSC	152.4	83.36	638.2	82.45	-	-	-	-	-	-
KrSSC	**7.3**	81.25	**18.7**	81.17	**192.4**	79.42	**411.5**	76.15	**683.2**	73.34
LRR	145.5	85.75	614.8	85.14	-	-	-	-	-	-
KrLRR	**7.1**	83.24	**16.4**	83.20	**160.8**	81.52	**384.5**	79.21	**641.5**	76.53
TRR	113.2	90.28	476.4	89.78	-	-	-	-	-	-
KrTRR	**6.5**	88.95	**15.8**	88.65	**168.2**	85.76	**403.8**	83.26	**795.6**	81.53
NVR3	118.5	91.85	531.1	91.28	-	-	-	-	-	-
KrNVR3	**8.3**	90.08	**22.5**	90.14	**243.6**	86.27	**627.5**	83.87	**968.4**	82.41

Table 3. The subspace clustering performance on the synthetic dataset.

No. Points	500		5000		10000		50000		100000	
	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.
SSC	135.4	94.15	1824	93.86	5413	91.05	-	-	-	-
KrSSC	**6.2**	92.12	**53.4**	91.18	**164.2**	89.73	**231.5**	85.04	**285.7**	81.85
LRR	118.6	95.27	1645	94.57	4853	92.14	-	-	-	-
KrLRR	**6.0**	93.24	**49.3**	92.21	**152.7**	89.49	**216.2**	86.03	**274.3**	82.20
TRR	89.5	98.85	1627	97.15	5825	95.69	-	-	-	-
KrTRR	**5.9**	98.06	**46.7**	96.53	**185.3**	95.05	**250.3**	93.16	**314.2**	89.06
NVR3	96.4	99.91	1752	98.61	6024	97.10	-	-	-	-
KrNVR3	**6.0**	99.07	**52.8**	98.11	**207.5**	96.24	**260.1**	93.89	**321.5**	90.62

Table 4. The average running time and clustering accuracy of our methods with different k.

k	2	3	4	5
average running time (seconds):				
KrSSC	715.6	285.7	61.2	25.4
KrLRR	682.5	274.3	52.7	20.6
KrTRR	755.1	314.2	84.3	31.5
KrNVR3	794.3	321.5	91.6	36.2
average clustering accuracy:				
KrSSC	83.14	81.85	75.42	67.25
KrLRR	84.43	82.20	77.16	68.17
KrTRR	90.75	89.06	84.27	73.41
KrNVR3	92.54	90.62	85.34	75.24

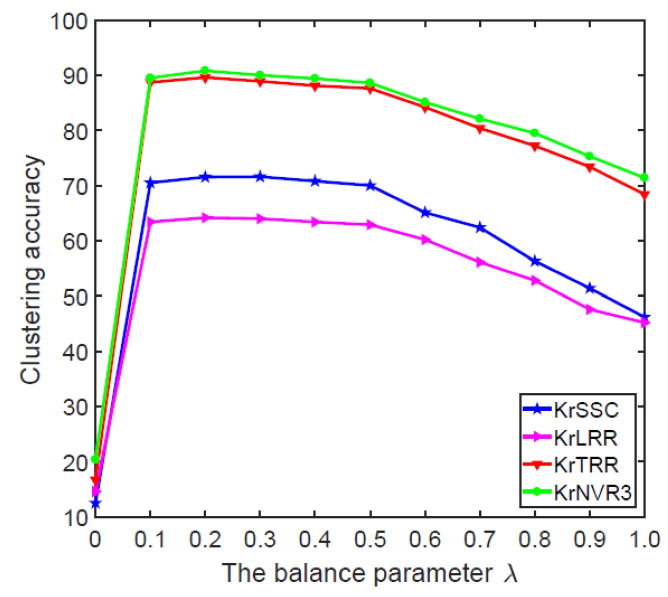

Fig.2 The average clustering accuracy with different balance parameter λ.

Conclusion

We have presented a fast subspace clustering model based on the Kronecker product. Due to the property that the Kronecker product of a block diagonal matrix and any other matrix is still a block diagonal matrix, we learn the representation matrix of spectral clustering using the Kronecker product of a set of smaller matrices. Thanks to the superiority of the Kronecker product in reducing the computational complexity of matrix operations, the memory space and computational complexity of our methods achieve significant efficiency gain compared with several baseline approaches (SSC, LRR, TRR, and NVR3).