International Journal of Electrical and Computer Engineering (IJECE)
Vol. 13, No. 6, December 2023, pp. 7115~7124
ISSN: 2088-8708, DOI: 10.11591/ijece.v13i6.pp7115-7124

Visualization of hyperspectral images on parallel and distributed platform: Apache Spark

Abdelali Zbakh[1], Mohamed Taj Bennani[2], Adnan Souri[3], Outman El Hichami[4]
[1]National School of Business and Management Tangier (ENCGT), ER-MSI, Abdelmalek Essaâdi University, Tetouan, Morocco
[2]LPAIS Laboratory, Computing Science Department, Faculty of Sciences Dhar El Mahraz Fez, Sidi Mohamed Ben Abdellah University, Fez, Morocco
[3]New Technology Trends for Innovation, Faculty of Sciences Tetouan, Abdelmalek Essaâdi University, Tetouan, Morocco
[4]Applied Mathematics and Computer Sciences Team, Higher Normal School, Abdelmalek Essaadi University, Tetouan, Morocco

Article Info

Article history:

Received Jan 2, 2023
Revised May 3, 2023
Accepted Jun 4, 2023

Keywords:

Dimension reduction
Hyperspectral
MapReduce
Principal component analysis
Spark platform
Visualization

ABSTRACT

The field of hyperspectral image storage and processing has undergone a remarkable evolution in recent years. The visualization of these images represents a challenge as the number of bands exceeds three bands, since direct visualization using the trivial system red, green and blue (RGB) or hue, saturation and lightness (HSL) is not feasible. One potential solution to resolve this problem is the reduction of the dimensionality of the image to three dimensions and thereafter assigning each dimension to a color. Conventional tools and algorithms have become incapable of producing results within a reasonable time. In this paper, we present a new distributed method of visualization of hyperspectral image based on the principal component analysis (PCA) and implemented in a distributed parallel environment (Apache Spark). The visualization of the big hyperspectral images with the proposed method is made in a smaller time and with the same performance as the classical method of visualization.

Corresponding Author:

Abdelali Zbakh
National School of Business and Management Tangier (ENCGT), ER-MSI, Abdelmalek Essaâdi University
Tetouan, Morocco
Email: a.zbakh@uae.ac.ma

1. INTRODUCTION

Currently, digital display devices produce a color image for the human eye using a combination of three primitive colors. So, a classic red, green and blue (RGB) color image is a combination of three layers (bands): RGB. A hue, saturation and lightness (HSL) color image is a combination of: HSL [1]. On the contrary, we find hyperspectral images composed of hundreds of layers (bands). A hyperspectral image can be described as a three-dimensional data cube consisting of two spatial dimensions and one spectral dimension. In this representation, each pixel contains a spectrum of wavelengths within the visible-near infrared range, spanning from 400 to 1,400 nanometers.

Hyperspectral imaging is frequently used in the field of remote sensing, environment monitoring [2], [3], polarimetric imaging, land cover classification [4], [5] and multimodal medical imaging. In astronomy, for example, hyperspectral imagery is used to archive soil and space observations. In medical imaging, hyperspectral imaging is used for the detection of diseases such as cancer [6].

Hyperspectral imaging produces high-dimensional data where each pixel in the image is represented by a spectrum of measurements across many different wavelengths. However, the high dimensionality of this data can make it challenging to analyze and interpret. Now, how to visualize a hyperspectral cube and give the user, not usually specialist, a synthetic view of the data contained in the image with the minimum possible loss,

and facilitate the interpretation of the image. Among the first solutions proposed is to visualize all the cube in the form of a video sequence, each layer of the cube is represented by an image. However, when we work in the plan and with a lot of hyperspectral images of large spectral dimensions, this solution remains difficult to practice.

So, to visualize a hyperspectral image in color and in the plan with the number of spectral bands which exceeds three bands, it is often necessary to reduce the dimensionality of hyperspectral images and obtain, from the original image, a composite image that consists of three bands. Dimension reduction techniques such as principal component analysis (PCA) or t-distributed stochastic neighbor embedding (t-SNE) can be employed to transform the high-dimensional hyperspectral data into a lower-dimensional space while preserving as much of the information as possible. By reducing the number of dimensions, it becomes easier to visualize and interpret the data. By visualizing the reduced-dimensional data, researchers and analysts can gain insights into the underlying patterns and relationships in the data, which can help with tasks such as identifying and classifying different materials or objects in the scene. This, in turn, can have applications in fields such as remote sensing, agriculture, and environmental monitoring. Several methods of visualizing hyperspectral images exist: Methods based on spectral band selection [7]–[9], methods based on weighting [5], methods based on optimization [6] and projection-based methods [10].

To bypass the computational problem posed by the processing of large hyperspectral cube [11], [12]. We will use an open source framework named Apache Spark [13], which distributes data storage in memory random-access memory (RAM) and processes the data in parallel. This choice has given us a considerable gain in the time of visualization of a hyperspectral image.

The rest of this paper is structured in the following manner: in section 2, we will present the related work on the hyperspectral image visualization methods. Next, in section 3, we will describe our parallel distributed visualization approach based on the PCA projection method. In section 4, we will experiment our approach on several free hyperspectral images. And we finish the paper with a conclusion and perspectives.

2. RELATED WORKS OF HYPERSPECTRAL IMAGE VISUALIZATION METHODS

In the field of hyperspectral imaging, there are several methods used to visualize a hyperspectral image. The literature review revealed that the four main methods used for this purpose are based on band selection, weighting, optimization, and transformation. The first method is based on band selection [7]–[9]. To visualize a hyperspectral image in an RGB representation system, three spectral bands must be selected from the original hyperspectral image composed of hundreds of bands. Then, each band will be assigned to a color: red, green and blue. This type of visualization method is used in the AVIRIS browser [14]. The visualization with this method is fast, but we take, just the existing data in the three selected bands and the data from other bands will be ignored. So, a large amount of existing information in the image is lost. The second method is based on weighting [15]. This method provides an image resulting from a linear combination of the input image bands. In this method there are two types: Method based on stretched color matching functions (CMFs) and Method based on bilateral filtering. The advantage of this method is the use of all the bands of the image, but the problem arises in the choice of the same weight that will be attributed to the pixels of the image by ignoring the variety of pixels. The third method is based on optimization [16]. In this method, some functions are applied to the image according to the optimized criterion. We find: Method based on Markov random field and Method based on multi maximization goals. The big challenge for this method is how to find the right function to apply it to the image. The last method is based on transformation. With this method, we can visualize a hyperspectral image by projecting the original image on a smaller dimension (three for example). Over the past few years, several techniques of dimensionality reduction have emerged to decrease the dimensionality of hyperspectral data to a lower-dimensional space, important examples include: Hessian eigenmap embedding, locally linear embedding (LLE), isometric feature mapping (ISOMAP), Laplacian eigenmap embedding, diffusion maps, conformal maps, independent component analysis (ICA) [17] and PCA [18], [19].

In this paper, we will use the PCA algorithm of the last method to do the visualization. PCA is among the dimension reduction algorithms that can be implemented effectively and which is used successfully in commercial remote sensing applications [20]. Since we are visualizing a large hyperspectral image, PCA does a lot of computation time [11], [12]. So, to solve this problem, we will use a distributed and parallel computing.

At present, there exist two widely-used libraries that offer a parallel distributed implementation of the PCA algorithm: MLlib on spark [21], [22] and the Mahout based on MapReduce [23]. Elgamal [24] demonstrated that these two libraries do not allow a flawless analysis of a large mass of data and introduced a novel implementation of PCA called sPCA. This proposed algorithm exhibits superior scalability and

accuracy compared to its competitors. Wu *et al.* [25] proposed a new distributed parallel implementation for the PCA algorithm. The implementation is done using the Spark platform and the results obtained are compared with a serial implementation on MATLAB and a parallel implementation on Hadoop. The comparison demonstrates the effectiveness of the proposed implementation in terms of both precision and computation time.

3. THE PROPOSED VISUALIZATION APPROACH

To comprehend the information concealed within the hyperspectral image cube or extract a relevant portion of the image, visualization is often employed. However, due to the limitations of human perception, we can only visualize a limited number of hyperspectral bands (typically up to 3). Before embarking on the visualization of our hyperspectral image, it is necessary to reduce the number of spectral bands to 3 without compromising the quality of information. In the subsequent steps, we will employ PCA, a widely used technique in various domains such as dimensionality reduction, image processing, data visualization, and discovering underlying patterns within the data.

3.1. Classic PCA algorithm

PCA [26] is a dimensionality reduction technique employed to reduce the dimensions of a matrix containing quantitative data. This approach enables the extraction of the dominant profiles from the matrix. To utilize the PCA algorithm (refer to Algorithm 1) on the hyperspectral image, we consider the hyperspectral image M as a matrix of size ($m = L \times C, N$), where C represents the number of columns, L represents the number of rows, and N represents the number of bands in the image. It is important to note that $m >> N$, indicating a significantly higher number of pixels than the number of spectral bands. Every row of the matrix M corresponds to a pixel vector. For example, the first pixel is represented by the vector: $[M11, M12, \dots, M1N]$, with $M1j$ is the value of the pixel 1 taken by the spectrum of number j. Each column of the matrix M represents the values of all pixels in the image captured by a specific spectrum. For example, $Mi1 = [M11, M21, \dots, Mm1]$ represents the data of the image taken by the spectrum 1. In the formula (1), $\overline{Mj}$ denoted the average of column j and σj denoted the standard deviation of column j. In the formula (2), $MRC^T.MRC$ denoted the matrix product between the transpose of the matrix MRC and the matrix MRC.

Algorithm 1. Classical PCA algorithm

```
Algorithm Classical_PCA(M)
Input: matrix M of dimension (m, N)
Output: matrix U of dimension (m,3)
# Calculate the reduced centered matrix of M denoted: MRC
   – for each i = 1…m and for each j = 1…N
```

$$MRC_{ij} = \frac{\mathrm{Mij} - \overline{\mathrm{Mj}}}{\sigma j} \quad (1)$$

With $\overline{Mj} = \frac{1}{m}\sum_{i=1}^{m} Mij$ and $\sigma j^2 = \frac{1}{m}\sum_{i=1}^{m}(Mij - \overline{Mj})^2$

```
# Calculate the correlation matrix of size (N, N) denoted: MC
```

$$MC = \frac{1}{\mathrm{m}}(\mathrm{MRC}^{\mathrm{T}}.\mathrm{MRC}) \quad (2)$$

```
   – Calculate the eigenvalues and eigenvector of the MC matrix denoted:
```

$$[\boldsymbol{\lambda}, \boldsymbol{V}]$$

```
   – Sort the eigenvector in descending order of the eigenvalues and take the first k
     columns of V(k < N)
   – Project the matrix M on the vector
```

$$\boldsymbol{V}: \boldsymbol{U} = \boldsymbol{M}.\boldsymbol{V}$$

```
   – use the new matrix U of size (m, k) for the visualization of the hyperspectral image
   – return U
```

3.2. Proposed distributed and parallel PCA algorithm

Due to the large size of hyperspectral images, the traditional PCA algorithm necessitates computationally intensive processing. In this section, we will introduce a parallel distributed implementation method of the algorithm utilizing the Spark platform. Given that a hyperspectral image captures the same scene across multiple spectral bands, we can decompose the hyperspectral image into individual images,

each representing a specific spectrum (as depicted in Figure 1). First, we start by transforming the hyperspectral cube of the image into a one-dimensional V vector of size N. Each element of V contains an image of size $L \times C$ according to a certain band Figure 1. Now, each *Vt* image, will be stored in the memory RAM as a resilient distributed data set (RDD). To make a parallel distributed implementation of PCA, we will leverage the map-reduce paradigm of Spark. The proposed algorithm operates in the following manner:

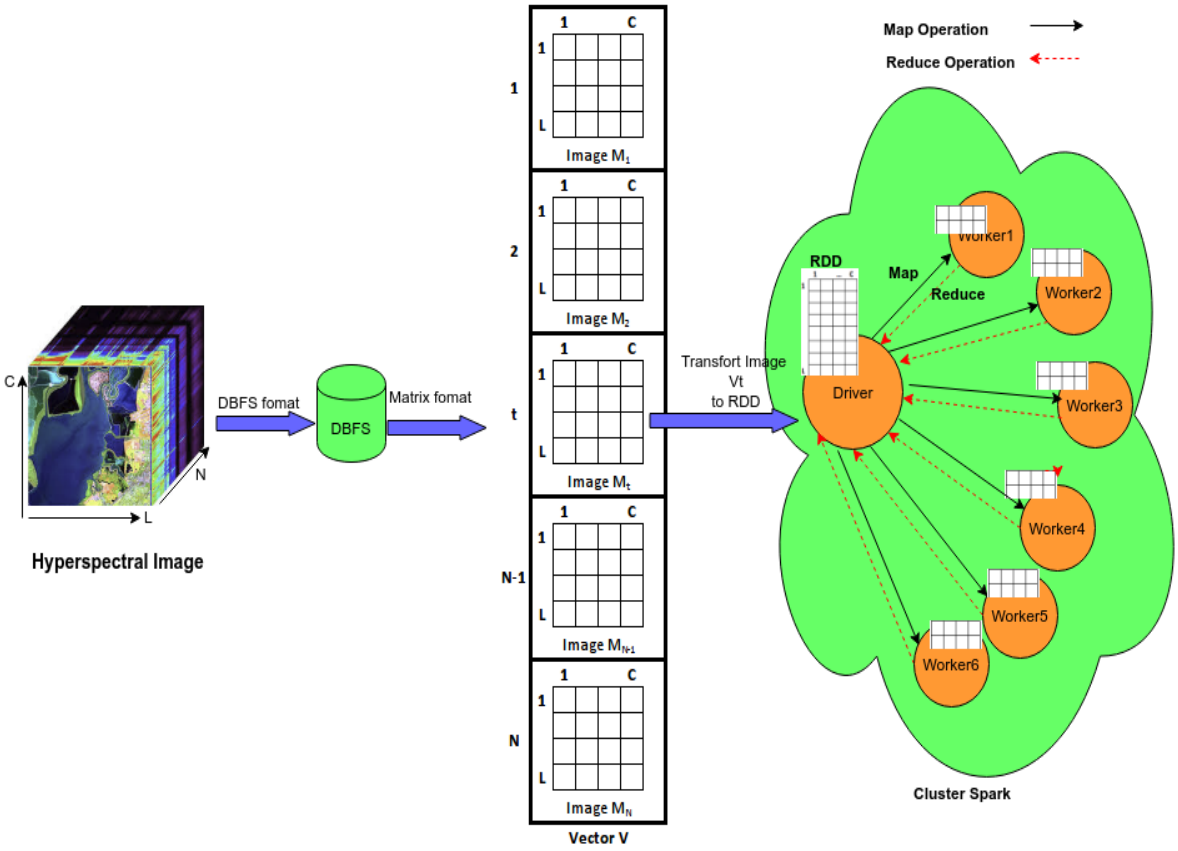


Figure 1. Descriptive diagram of the proposed PCA algorithm

Step 1: Calculate the reduced centered matrix of V:

As previously mentioned, the vector V comprises multiple images, with each image *Vt* represented by a matrix (referred to as a resilient distributed data in Spark notation) of size (*L*, *C*), where *L* represents the number of rows in the image and *C* represents the number of columns. Hence, in order to compute the reduced centered matrix of V, denoted as *MRC*, a parallel distributed computation is performed on each image *Vt*.

– Calculate the centered matrix of V denoted *MC* algorithm 2:

$$MC_t ij = V_t ij - \overline{V}_t \text{ for each } i = 1 \text{ to } L \text{ and for each } j = 1 \text{ to } C \quad (3)$$

with

$$\overline{V}_t = \sum_{i=1}^{L} \sum_{j=1}^{C} \left(\frac{1}{LxC} x\ V_t ij\right)$$

Algorithm 2. Calculating the centered matrix with Spark

```
Algorithm Center_Images (V)
Input: vector images V
Output: vector of centered images MC
        For each image Vt do
```

```
Map1:
        for each line i of image Vt do
                calculate X[i]=sum (Vti)
        return X

Reduce1:
        calculate the average of X denoted avg

Map2:
        for each line i of image Vt
           for each value Vti,j
                  calculate MC tij= Vtij-avg
        return MC
```

In the formula 3, $\overline{V_t}$ denoted the average of image V_t

– Calculate the reduced centered matrix of M denoted MRC algorithm 3:

$$MRC_t ij = \frac{\mathrm{MC_t ij}}{\sigma_t} \text{ for each } i = 1 \text{ to } L \text{ and for each } j = 1 \text{ to } C \quad (4)$$

with

$$\sigma_t{}^2 = \frac{1}{LxC}\sum_{i=1}^{L}\sum_{j=1}^{C}(V_t ij - \overline{V_t})^2$$

or

$$\sigma_t{}^2 = \frac{1}{LxC}\sum_{i=1}^{L}\sum_{j=1}^{C}(MC_t ij)^2$$

In the formula 4, σ_t denoted the standard deviation of image V_t.

Algorithm 3. Calculating the reduced centered matrix with Spark

```
Algorithm Reducing_Images (MC)
Input: vector images V centered
Output: the reduced centered matrix MRC

For each image centered MC_t do

Map1:
# Calculate the standard deviation of image Vt denoted σt
        for each line i of image MC_t do
        for each value MC_t i,j do
              calculate MC_t i, j = (MC_t i,j * MC_t i,j)/(LxC)
        return MC_t

Reduce1:
        Calculate sqrt(sum(MC_t)) denoted σt

Map2:
        for each line i of image MC_t do
                for each value MC_t i,j do
                   calculate MRC_t ij = MC_t i,j/σ_t

        return MRC
```

Step 2: Calculate the *MRC* correlation matrix of size (*N*, *N*) denoted: $MCoor$

As stated in step 1, the *MRC* represents an image vector of size N. Every image MRC_t corresponds to a reduced and centered matrix. We will now utilize Spark's distributed parallel computation framework, MapReduce, to compute the correlation matrix of size (*N*, *N*) by performing a matrix product operation between the MRC^T and *MRC* vectors.

$$MCorr = \frac{1}{LxC}(MRC^T . MRC) \quad (5)$$

$$MCorr_{t,k} = \frac{1}{\mathrm{LxC}}(\mathrm{MRC_t} . \mathrm{MRC_k}) \text{ for each } t = 1 \text{ to } N \text{ and for each } k = 1 \text{ to } N \quad (6)$$

with

$$MRC_t \,.\, MRC_k = \sum_{i=1}^{L} \sum_{j=1}^{C} MRC_t ij \,.\, MRC_k ij$$

To determine the value of each $MCorr_{t,k}$ in formula (6), the image MRC_t is multiplied by the image MRC_k pixel by pixel. Then we calculate the average of result algorithm 4:

Algorithm 4. Calculation of the correlation matrix using Spark

```
Algorithm Correlation_Images (MRC)
Input:  Reduced centered matrix MRC
Output: Correlation matrix Mcorr of size NxN
  For each 1 ≤ t ≤ N do
  For each 1 ≤ k ≤ N     do
        Calculate S = Product(MRC_t, MRC_k)
        MCorr_{t,k} = S/(L × C)
  return MCorr

Algorithm Product(MRC_t, MRC_k)
Input: 2 Reduced centered matrix MRC_t, MRC_k
Output: sum(the product of two images pixel by pixel)
Map1:
        for each line i of image MRC_t do
                calculate X_i = (MRC_t i, MRC_k i)
        return X

Map2:
        for each line i of image X do
                calculate X_i= scalar product between X_i[0] et X_i[1]
        return X

Reduce1:
        Calculate Sum(X) denoted S
        Return S
```

Step 3: Calculate the eigenvector and eigenvalues of the MCorr matrix: $[\lambda, V]$

Step 4: Arrange the eigenvectors in descending order based on their corresponding eigenvalues and select the first three columns of V ($3 < N$)

Step 5: Perform a projection of the matrix M onto the vector V: $U = M.V$

Step 6: Utilize the newly obtained matrix U, which has a size of (m, 3), for the purpose of visualizing the hyperspectral image.

4. EXPERIMENTS AND COMPUTATIONS

To test the proposed algorithm, the free visible air infra-red imaging spectrometer (AVIRIS) Moffett Field image was used with 224 spectral bands in the 2.5 nanometers to 400 nanometers, which was acquired on August 20, 1992 [14]. On this hyperspectral image, we took samples of different sizes, see Table 1, and then on each image obtained, we tested the proposed distributed parallel algorithm and a serial implementation of classical PCA from the Python library scikit-learn. We collected the three most significant eigenvalues as shown in Table 2 and the execution time of each algorithm as shown in Figure 2.

Table 1. Datasets

	Name of dataset	Spatial dimensions	Hyperspectral bands
Dataset 1	Moffett Field	500×500	3
Dataset 2	Moffett Field	500×500	10
Dataset 3	Moffett Field	1924×753	3
Dataset 4	Moffett Field	1924×753	10
Dataset 5	Moffett Field	1924×753	15
Dataset 6	Moffett Field	1924×753	20
Dataset 7	Moffett Field	1924×753	25
Dataset 8	Moffett Field	1924×753	50
Dataset 9	Moffett Field	1924×753	75
Dataset 10	Moffett Field	1924×753	100
Dataset 11	Moffett Field	1924×753	150
Dataset 12	Moffett Field	1924×753	224

Table 2. Example of the top three eigenvalues obtained from PCA

	Sklearn PCA	Proposed PCA
Dataset 1	1.9371026343	1.93710263
	0.913755533084	0.9137555
	0.149141832615	0.14914183
Dataset 5	10.06478855	10.06478855,
	4.01771578	4.01771578,
	0.5480712	0.5480712
Dataset 12	160.63876264	160.63876264,
	28.00313352	28.00313352,
	14.56390331	14.56390331

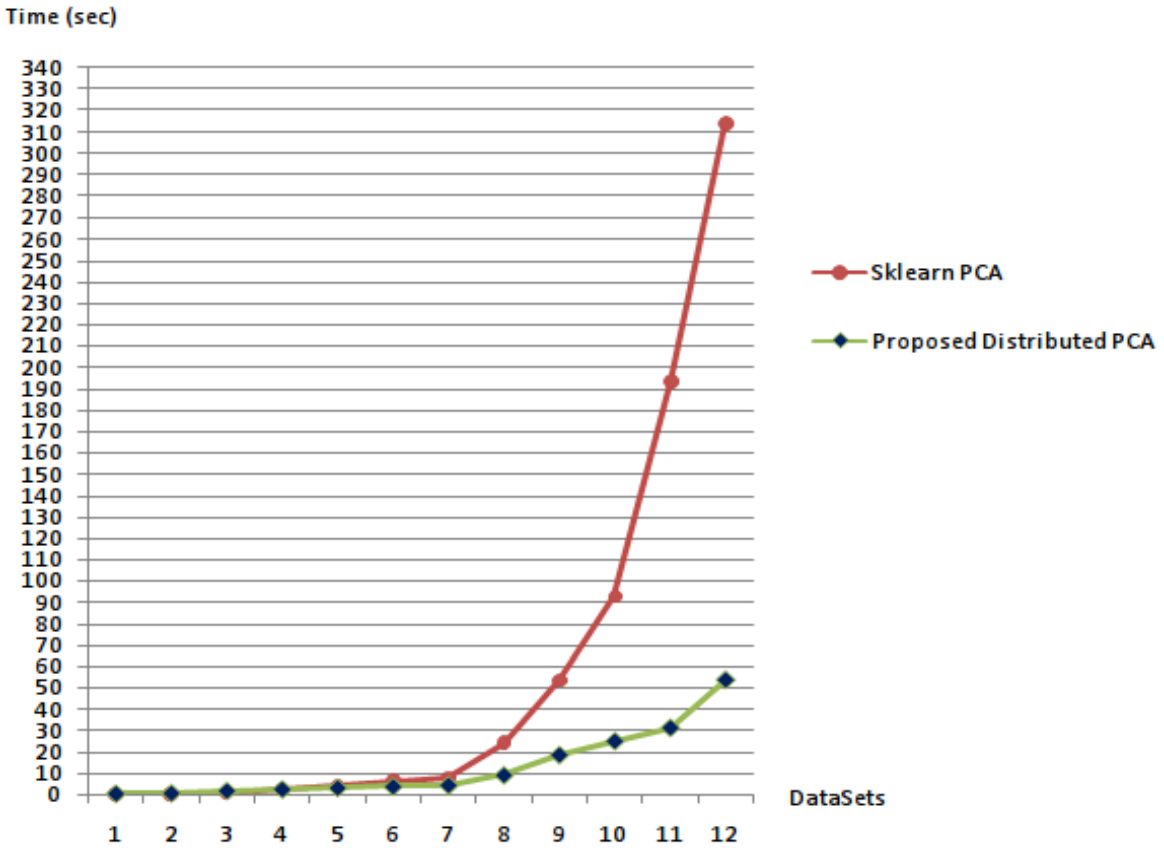


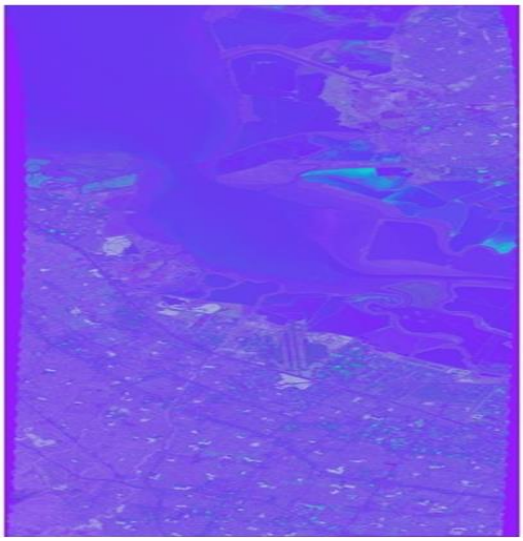
Figure 2. The runtime comparison between sklearn and the proposed PCA

The classical PCA of the scikit-learn library is tested on one computer equipped with: CPU: Intel® Core ™ i7-2820QM CPU @ 2.30 GHz × 8, RAM: 8G, OS: Ubuntu 16.04 LTS. The proposed distributed parallel algorithm has been tested on the cloud Databricks [26] of the configuration, as shown in Table 3. Both algorithms are programmed with the Python language. The runtime comparison of the two algorithms shows the speed of PCA sklearn for small images, but if the image has a high number of bands, more than 10 spectral bands our proposed PCA is faster, see Figure 2. Figure 3 illustrates the visualization of a hyperspectral image dataset. In Figure 3(a), the image is displayed without any PCA applied, presenting the original representation. In contrast, Figure 3(b) showcases the image after the application of either classical PCA or the newly proposed PCA distributed method.

Table 3. Configuration parameters of cluster spark in cloud Databricks

Driver	Nodes
Driver type: 36 GB memory, 8 Cores	Number of Worker Nodes: 6 For each worker :8.0 GB Memory, 2 Core

(a)



(b)

Figure 3. Visualization of hyperspectral image (dataset 12) in (a) the image is displayed without PCA, while on the (b) the image is shown after applying either classical PCA or the proposed PCA

5. CONCLUSION

In this work, a method of visualizing a hyperspectral image has been proposed based on the reduction of the dimensionality of the image in a parallel distributed environment. The algorithm has been developed using Python 3, and evaluated on hyperspectral images utilizing the Spark platform. The results obtained align with those of traditional PCA, and the visualization of the images post-application of our reduction algorithm confirms the validity of our algorithm. By comparing the execution time of the two algorithms: sklearn PCA and proposed PCA, we discovered that the proposed PCA algorithm displays faster performance when processing large images. This observation implies that the proposed PCA algorithm may be more efficient and effective in handling larger data sets compared to the classic algorithm.

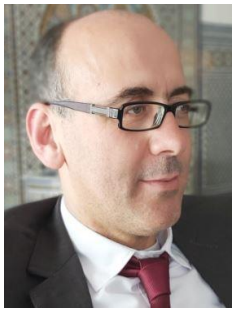
REFERENCES

[1] J. S. Tyo, A. Konsolakis, D. I. Diersen, and R. C. Olsen, "Principal-components-based display strategy for spectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 3, pp. 708–718, Mar. 2003, doi: 10.1109/TGRS.2003.808879.

[2] J. Pontius, M. Martin, L. Plourde, and R. Hallett, "Ash decline assessment in emerald ash borer-infested regions: A test of tree-level, hyperspectral technologies," *Remote Sensing of Environment*, vol. 112, no. 5, pp. 2665–2676, May 2008, doi: 10.1016/j.rse.2007.12.011.

[3] Y.-T. Chan, S.-J. Wang, and C.-H. Tsai, "Real-time foreground detection approach based on adaptive ensemble learning with arbitrary algorithms for changing environments," *Information Fusion*, vol. 39, pp. 154–167, Jan. 2018, doi: 10.1016/j.inffus.2017.05.001.
[4] G. Cheng, J. Han, and X. Lu, "Remote sensing image scene classification: Benchmark and state of the art," *Proceedings of the IEEE*, vol. 105, no. 10, pp. 1865–1883, Oct. 2017, doi: 10.1109/JPROC.2017.2675998.
[5] P. Duan, X. Kang, S. Li, and P. Ghamisi, "Noise-robust hyperspectral image classification via multi-scale total variation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 6, pp. 1948–1962, Jun. 2019, doi: 10.1109/JSTARS.2019.2915272.
[6] Y. Khouj, J. Dawson, J. Coad, and L. Vona-Davis, "Hyperspectral imaging and K-means classification for histologic evaluation of ductal carcinoma in situ," *Frontiers in Oncology*, vol. 8, Feb. 2018, doi: 10.3389/fonc.2018.00017.
[7] H. Su, Q. Du, and P. Du, "Hyperspectral image visualization using band selection," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 2647–2658, Jun. 2014, doi: 10.1109/JSTARS.2013.2272654.
[8] Yuan Yuan, Guokang Zhu, and Qi Wang, "Hyperspectral band selection by multitask sparsity pursuit," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 2, pp. 631–644, Feb. 2015, doi: 10.1109/TGRS.2014.2326655.
[9] G. Zhu, Y. Huang, J. Lei, Z. Bi, and F. Xu, "Unsupervised hyperspectral band selection by dominant set extraction," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 1, pp. 227–239, Jan. 2016, doi: 10.1109/TGRS.2015.2453362.
[10] C. Theoharatos, V. Tsagaris, N. Fragoulis, and G. Economou, "Hyperspectral image fusion using 2-D principal component analysis," in *2011 2nd International Conference on Space Technology*, Sep. 2011, pp. 1–4, doi: 10.1109/ICSpT.2011.6064682.
[11] W. Liu, X. Yang, D. Tao, J. Cheng, and Y. Tang, "Multiview dimension reduction via hessian multiset canonical correlations," *Information Fusion*, vol. 41, pp. 119–128, May 2018, doi: 10.1016/j.inffus.2017.09.001.
[12] R. S. Lynch and P. K. Willett, "Use of Bayesian data reduction for the fusion of legacy classifiers," *Information Fusion*, vol. 4, no. 1, pp. 23–34, Mar. 2003, doi: 10.1016/S1566-2535(02)00098-2.
[13] S. Salloum, R. Dautov, X. Chen, P. X. Peng, and J. Z. Huang, "Big data analytics on apache spark," *International Journal of Data Science and Analytics*, vol. 1, no. 3–4, pp. 145–164, Nov. 2016, doi: 10.1007/s41060-016-0027-9.
[14] A. Zbakh, Z. Alaoui, A. Benyoussef, A. El, and M. El, "Spectral classification of a set of hyperspectral images using the convolutional neural network, in a single training," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 6, 2019, doi: 10.14569/IJACSA.2019.0100634.
[15] K. Kotwal and S. Chaudhuri, "Visualization of hyperspectral images using bilateral filtering," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 5, pp. 2308–2316, May 2010, doi: 10.1109/TGRS.2009.2037950.
[16] M. Mignotte, "A bicriteria-optimization-approach-based dimensionality-reduction model for the color display of hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 2, pp. 501–513, Feb. 2012, doi: 10.1109/TGRS.2011.2160646.
[17] Q. Du, N. Raksuntorn, S. Cai, and R. J. Moorhead, "Color display for hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, no. 6, pp. 1858–1866, Jun. 2008, doi: 10.1109/TGRS.2008.916203.
[18] B. Zhao, X. Dong, Y. Guo, X. Jia, and Y. Huang, "PCA dimensionality reduction method for image classification," *Neural Processing Letters*, vol. 54, no. 1, pp. 347–368, Feb. 2022, doi: 10.1007/s11063-021-10632-5.
[19] Xiuping Jia and J. A. Richards, "Segmented principal components transformation for efficient hyperspectral remote-sensing image display and classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 37, no. 1, pp. 538–542, 1999, doi: 10.1109/36.739109.
[20] H. Zhang, D. W. Messinger, and E. D. Montag, "Perceptual display strategies of hyperspectral imagery based on PCA and ICA," in *Proc. SPIE 6233, Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XI*, May 2006, doi: 10.1117/12.665696.
[21] X. Meng *et al.*, "MLlib: Machine learning in apache spark," *Journal of Machine Learning Research*, vol. 17, pp. 1–7, 2016.
[22] O. Azeroual and A. Nikiforova, "Apache spark and MLlib-based intrusion detection system or how the big data technologies can secure the data," *Information*, vol. 13, no. 2, Jan. 2022, doi: 10.3390/info13020058.
[23] N. S. Sagheer and S. A. Yousif, "A parallel clustering analysis based on hadoop multi-node and apache mahout," *Iraqi Journal of Science*, pp. 2431–2444, Jul. 2021, doi: 10.24996/ijs.2021.62.7.32.
[24] T. Elgamal, M. Yabandeh, A. Aboulnaga, W. Mustafa, and M. Hefeeda, "SPCA: Scalable principal component analysis for big data on distributed platforms," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, May 2015, pp. 79–91, doi: 10.1145/2723372.2751520.
[25] Z. Wu, Y. Li, A. Plaza, J. Li, F. Xiao, and Z. Wei, "Parallel and distributed dimensionality reduction of hyperspectral data on cloud computing architectures," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 6, pp. 2270–2278, Jun. 2016, doi: 10.1109/JSTARS.2016.2542193.
[26] Y. O. Sayad, H. Mousannif, and H. Al Moatassime, "Predictive modeling of wildfires: A new dataset and machine learning approach," *Fire Safety Journal*, vol. 104, pp. 130–146, Mar. 2019, doi: 10.1016/j.firesaf.2019.01.006.

BIOGRAPHIES OF AUTHORS



Abdelali Zbakh is an assistant professor in Computer Science at National School of Business and Management-Tangier, Abdelmalek Essaâdi University Morocco. He is a former Teacher of computing at preparatory classes for engineering schools – Tangier. He received his Ph.D. degree in computer science from the faculty of sciences Rabat - Mohammed V university of Rabat. His current research interests include information systems, machine learning and deep learning. He can be contacted at email a.zbakh@uae.ac.ma.

Mohamed Taj Bennani received his master's degree in Computer Science and Networking from Science Faculty of Tangier, Tangier in 2011. He received his Ph.D. in 2019 (Computing Science and Networking) from Science Faculty of Tangier. At present, he is working as a prof. at Faculty of Science of Dhar el Mahraz Fez since 2019. He can be contacted at email Bennani.taj@gmail.com.

Adnan Souri is an assistant professor in Computer Science at Abdelmalek Essaâdi University, Faculty of Sciences Tetouan, Morocco. He is a former Teacher of computing at preparatory classes for engineering schools – Tangier. He received his Ph.D. degree in computer science from the Faculty of Sciences Tetouan, Abdelmalek Essaâdi University Morocco. His current research interests include artificial intelligence, artificial neural networks and algorithms. Their current project is "Arabic language processing". He can be contacted at email a.souri@uae.ac.ma.

Outman El Hichami received his Ph.D. in 2017 from the Faculty of Sciences in Tetouan, Morocco. In 2018, he joined the Higher Normal School in Tetouan as an assistant professor in Computer Science, where he was promoted to associate professor in 2022. His research interests are in formal methods and machine learning. He can be contacted at email oelhichami@uae.ac.ma.