Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Topi

COMPLEX ENGINEERING PROBLEM

Course – ES304
Linear Algebra-II

Course Instructor
Muti – Ur - Rehman

Authors name Warda Bibi

Faculty of computer science and engineering
Ghulam Ishaq Khan Institue
swabi, Pakistan
Reg No. 2020517

Authors name Wardah Tariq

Faculty of computer science and engineering
Ghulam Ishaq Khan Institue
swabi, Pakistan
Reg No. 2020519

Authors name Zartaj Asim

Faculty of computer science and engineering
Ghulam Ishaq Khan Institue
swabi, Pakistan
Reg No. 2020526

Abstract

Principal Components Analysis is a common and widely used technique on large data types to help reduce the dimensionality by increasing the interpretability and preserve the data at the same time. This statistical technique for data decorrelation uses eigenvectors for object detection. Our paper briefly introduces PCA implementation on the multispectral image with high dimension to develop an understanding of PCA and how it helps in the processing of multidimensional image and its relationship with Linear Algebra-II.

The main aim of our assessment is to develop an understanding of Principle Component Analysis by implementing it on a multispectral image and perform the associated tasks using Python Programming Language.

I. INTRODUCTION

Principal Analysis Component can be described as a powerful method that belongs to linear transformations to be helpful in image processing. Following paper presents Principal Analysis Component as a technique for data compression, reduction of dimension and error analysis on the image.

Our assessment includes the following tasks that has been done.
Task 1 of cropping the dimension, concatenation of bands and visualization has been implemented on the multispectral image of Landsat ID "LC09_L2SP_150036_20220408_20220520_02_T1".

Task 2 has been implemented using 3 number of components by writing the code in Python Programming Language. We have implemented PCA using 3 number components on our image.
Lastly**, Task 3** is calculating error analysis on the use of Principal Analysis Component on the image and compare it with the original image using mean square error.

II. METHODOLOGY

The steps required for PCA are as follow:

A. Arrange data into normalized form by dimensionality reduction
B. Find the covariance of the standardized data
C. Calculate the eigenvalues and eigenvectors of the covariance matrix
D. Choose your principal components and form vectors
E. Plot the derived data

Task -- 1

Task 1 requires us to perform basic operations of visualizing different bands, cropping the image in different dimension and concatenating the bands. Using the gdal.open( ) function we have read our image and scaled the data from the scaleMinMax( ) function in order to standardize the observations such that the values should be between 0 and 1.
To visualize different bands we have used plt.figure( ) function as shown in **Fig1.1**. To extract each band and read it as a Matrix ds.GetRasterBand( ).ReadAsArray( ) is used. In order to perform the task of concatenation we have used a NumPy function of dtack() to concatenate all images upon each other and plotted as shown in **Fig1.2**.
Lastly to crop the original coordinates in new dimension we have used window function and gdal.Translate() to crop the .tiff image.

Old Coordinates

Upper Left (0.0,   0.0)
Lower Left (0.0, 1041.0)
Upper Right (1024.0,   0.0)
Lower Right (1024.0, 1041.0)
Center (512.0, 520.5)

New Coordinates

upper_left_x = 216
upper_left_y = 49
lower_right_x = 813
lower_right_y = 995

TASK -- 2

Using gdal.Open we have read our image and divided it into 3 bands in the form of a Matrix. The data is standardized using the scaling function as "scaleMinMax(x)" which is useful to find the maximum variance. Using the dstack() function the 3 bands "rMinMax,gMinMax,bMinMax" are concatenated in order to apply PCA. The function plt.figure() is used to plot the figure before applying PCA as **Fig1.4**.

The dimensions of each band is 1041 x 1024 as rows and columns respectively. To convert each column of 1024 to a vector we have used the reshape () function which results in 3 compressed vectors. The compressed vectors are stored in a variable as Matrix. Since our data has been standardized, we then calculated the covariance of our Matrix. The covariance of a p x p Matrix is defined as

$$cov_{x,y} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{N-1}$$

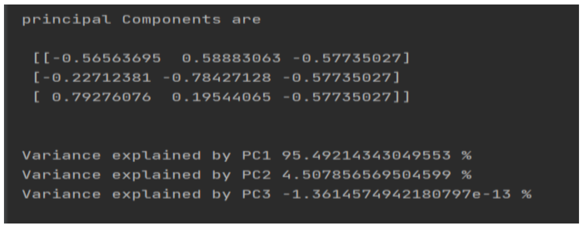
After calculating the covariance Matrix, the next step is to find out eigenvalues and eigenvectors. In our assessment we have used NumPy function of np.linalg to calculate the eigenvalues and eigenvectors. The eigenvectors of our covariance Matrix are the principal components. The first PCA is the highest eigenvalue of Matrix, similarly the highest eigenvalue of Matrix is the second PCA and so on. Since we have used 3 bands, we get three eigenvalues as PC1, PC2, and PC3 after sorting them. The final step before plotting is to transform our Matrix using the equation **Y = P^T X** , where Y is reduced Matrix and P^T is Principal Component and X is Matrix from original data. Lastly we use reshape( ) function to standardize our Matrix and plot it using plt.figure( ) **Fig1.5** and **Fig1.6**.

Task 3

Task 3 requires us to perform error analysis on the PCA image and compare the resulting image with the original image using mean square error.
To compute the variance of the PCA we use the following equation

$$\text{explained variance of } PC_k = \left(\frac{\text{eigenvalue of } PC_k}{\sum_{i=1}^{p} \text{eigenvalue of } PC_i} \right)$$

where PC$_k$ is the respective PCA whose variance is being calculated over the sum of all the eigenvalues. The following is the calculated variance.

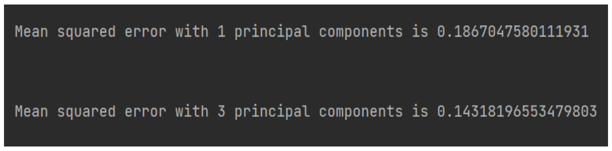


```
principal Components are

 [[-0.56563695  0.58883063 -0.57735027]
 [-0.22712381 -0.78427128 -0.57735027]
 [ 0.79276076  0.19544065 -0.57735027]]


Variance explained by PC1 95.49214343049553 %
Variance explained by PC2 4.507856569504599 %
Variance explained by PC3 -1.3614574942180797e-13 %
```

To calculate the mean square error MSE the following formula is used

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \tilde{y}_i)^2$$

where y is our original data Matrix and ȳ is our PCA Matrix.
Mean Square error with 1 Principal Component is 0.1867047580111931 whereas MSE with 3 Principal component is 0.14318196553479803.



```
Mean squared error with 1 principal components is 0.1867047580111931


Mean squared error with 3 principal components is 0.14318196553479803
```

III. RESULTS AND CONCLUSION

(Task 1)

Visualization of different bands

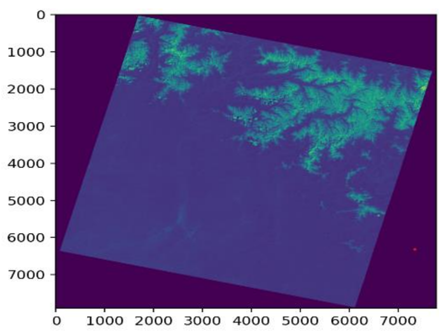


Fig 1.1

Concatenate the bands

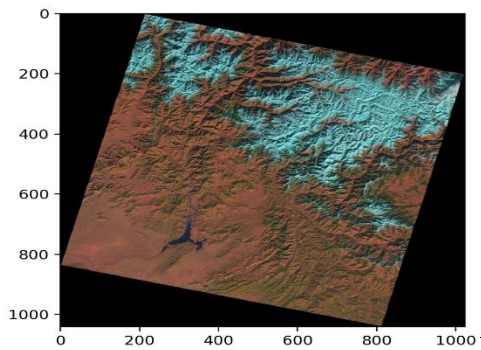


Fig 1.2

Cropping image in new dimension

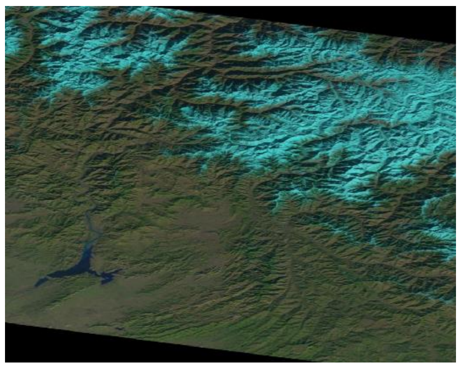


Fig 1.3

(TASK 2)

Before Implementing PCA

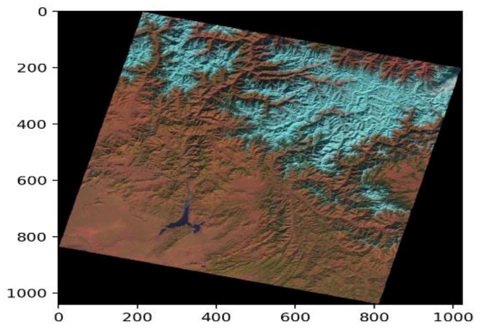


Fig 1.4

After Implementing PCA 1

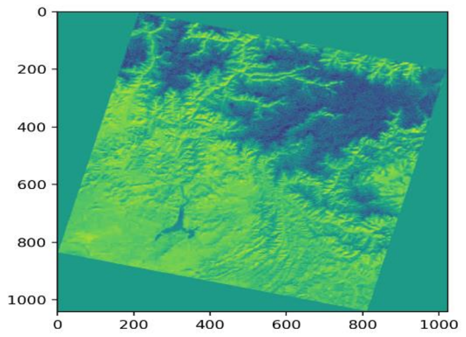


Fig 1.5

After Implementing PCA 2

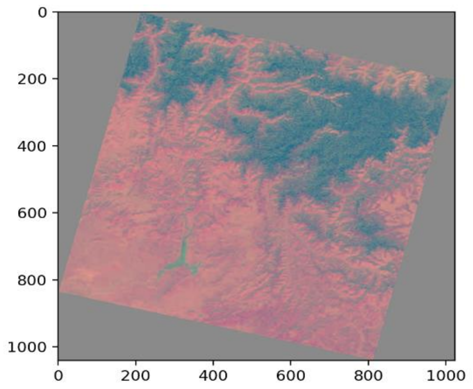


Fig 1.6

Task Distribution

We divided the code and report equally amongst ourselves.

Warda Bibi applied PCA on multispectral image. In report she wrote about task 2, methodology and conclusion.

Wardah Tariq performed basic operations like cropping, concatenating and visualizing bands. In report she wrote about task 1, references and task distribution.

Zartaj Asim performed an error analysis by comparing the original image with the images resulting from different number of principal components. She also wrote about abstract, introduction and task 3 in report.

References

[1] "Introduction to principal component analysis for image classification,".(2022, December 18) Retrieved from "https://www.youtube.com/watch?v=tU4Rm0Y_jQI"

[2] S. Kumar, "Principal component analysis: in-depth understanding through image visualization", in press.

[3] Image processing, (2022, December 18). Retrieved from " https://www.imageeprocessing.com/2017/11/principal-component-analysis.html"

[4] Detecting Multicollinearity Using Variance Inflation Factors, Penn State University, https://online.stat.psu.edu /stat462/node/180/

[5] Principal Component Analysis, UC Business Analytics R Programming Guide, https://uc-r.github.io/pca

[6] Python Data Science Handbook, Jake VanderPlas, O'Reilly, https://www.oreilly.com/library/view/python-data-science/9781491912126/

[7] Principal Component Analysis and Optimization: A Tutorial, Robert Reris and J. Paul Brooks, https://pdfs.semanticscholar.org/e0be/f0bd8e07de281230ae5df28daabb4047e8f0.pdf