Image Compression Using Singular Value Decomposition

Richa Patel[1]
Masters of technology in Data Science
Ahemdabad University
Ahemdabad,Gujarat,India
pricha9265@gmail.com

Shakirali Vijapura[2]
Masters of technology in Data Science
Ahemdabad University
Ahemdabad,Gujarat,India
shakiralivajapura97@gmail.com

***Abstract*—In data compression the actual image is transformed with some bits which is known as Image compression. To send or record the data of image in an effective manner, it is required to decrease the irrevalance and redundancy of the image data. The speed of transmission increases and the time taken to transmit the image data in the network decreases when image is compressed. The data of image is lost during the process of compression in the technique of lossy compression and viceversa in lossless compression technique. Multiple image compression techniques are available, which are used to solve these kind of issues. How image compression is performed and what kind of technology is used? These questions may pops up. So there exists two types of techniques: Lossy image compression and lossless image compression. These techniques consumes very less memory and are easy to apply. This system proposes to use lossy compression using SVD. The concept of Singular Value Decomposition is thoroughly explained and implemented on the image and results of image measures are presented accordingly.**

***Index Terms*—Singular Value Decomposition, Image Processing,Image Compression**

I. Introduction

A. Image Compression

Eliminating the irrelevance or redundancy in the data of image is the purpose of the compression techniques to store or transmitthe data in effective manner. The initial step is to transform the image from the representation of their spatial domain into a separate type of the representation by the use of few already known conversions and then encodes the converted values i.e., coefficients. As compared to predictive techniques, this techniques provides much better compression of the image data. But the computational cost is pretty high. Below mentioned three basic data redundancies which has to removed in order to be able to compress the image:

1. Redundancy in coding: When the code words which has the smallest length are used then this redundancy in coding occurs.

2. Redundancy in inter-pixel: When two images are correlated in terms of pixel then this redundancy occurs.

3. Redundancy in psycho-visual: The data of image that is basically not considered by visual system of human is the cause of this redundancy.

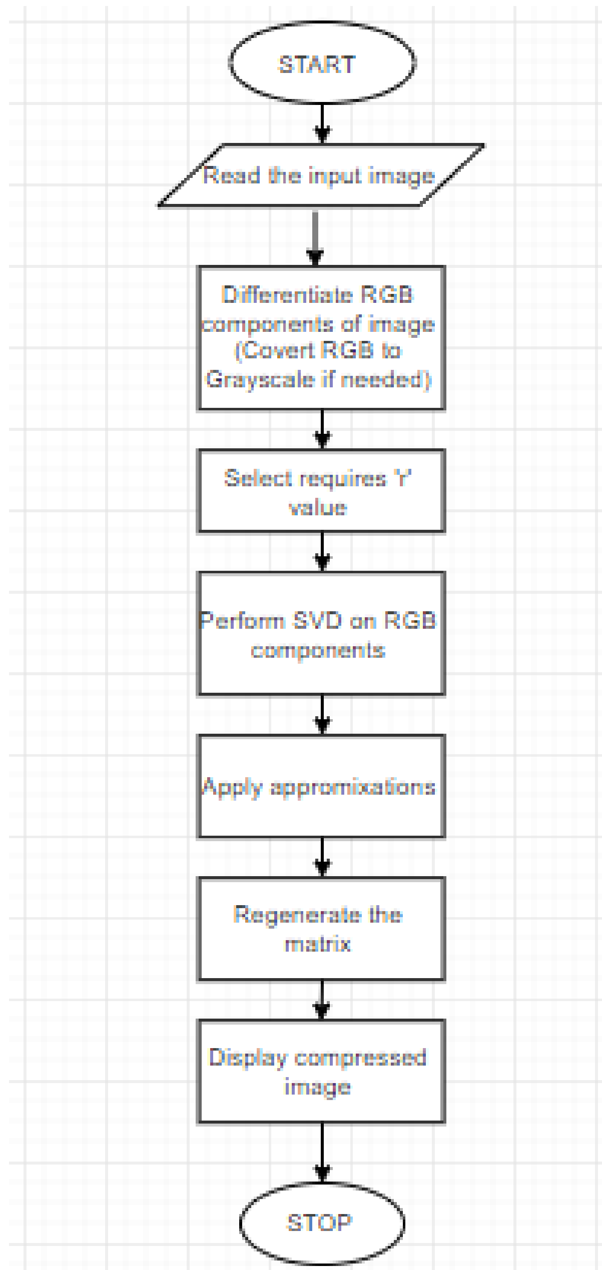

Fig. 1. Image Compression Process

B. Need of Compression

- Increased enough amount of the storage space.
- Decrease the transmission time of an image to get sent on the internet or get downloaded from webpages.
- Multimedia Applications: Desktop Editing
- Image Archiving: Data from Satellite.
- Image Transmission: Data from the web.

C. Image Compression Techniques

The image compression technique can be defined as the technique which reduces the storage space required to store the image but image quality does not degrade. This techniques are classified in two types:

- Lossy technique of image compression:

In lossy technique of image compression when the image is being compressed it cannot be reconstructed and there is loss of data while compression. When the a small loss of data in image is acceptable, this type of compression is most suited. From all the compression techniques used, most of them are lossy compression techniques. SVD is a lossy technique of image compression.

- Lossless technique of image compression:

Lossless image compression is a reversible process. Once constructed, it can be reconstructed. The compressed image and the original image both are same. Applications like medical are best suited for this technique.

II. Literature Review

Plenty of research has already been done in the area of image compression. For efficient compression and process of reconstruction, SVD is used as the application of linear algebra. For retrieval of color and even gray-scale image compression, this techniques is used which is mentioned in [3]. Variable rank matrix SVD algorithms and variable quantization matrix DCT are used for various class of images. Both of them are compared on performace which is mentioned in [4]. After the image is compressed, it is very important to improve the quality of image during the retrieval phase, there are certain techniques available which are mentioned in [5].

III. Theory of SVD (Singular Value Decomposition)

A. Singular Value Decomposition Process

Many mathematicians have stated that the most significant and the important topic in the area of linear algebra is SVD. There are many important features of SVD, one of them is that it can decompose into basic matrix. SVD can be applied on any mxn matrix that has be real matrix. Suppose we have a matrix A with m rows and n columns, with r(rank of matrix) and $r \leqslant m \leqslant n$. Then matrix A can be decomposed into other three matrices in which one matrix is diagonal matrix and other two are orthogonal matrix:[6]

$$A = U\Sigma V^T = \underbrace{[\mathbf{u}_1 \quad \mathbf{u}_2 \quad \dots \quad \mathbf{u}_r}_{\text{Col } A} \quad \underbrace{\mathbf{u}_{r+1} \quad \dots \quad \mathbf{u}_m]}_{\text{Nul } A^T} \begin{bmatrix} \sigma_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 & 0 & \dots & 0 \\ & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \sigma_r & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \dots \\ \mathbf{v}_r^T \\ \mathbf{v}_{r+1}^T \\ \dots \\ \mathbf{v}_n^T \end{bmatrix} \begin{matrix} \left.\vphantom{\begin{matrix} a \\ a \\ a \\ a \end{matrix}}\right\} \text{Row } A \\ \left.\vphantom{\begin{matrix} a \\ a \\ a \end{matrix}}\right\} \text{Nul } A \end{matrix} \tag{1}$$

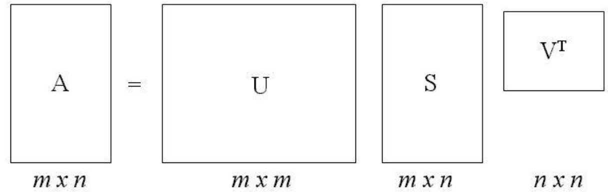

Fig. 2. Decomposing A to USV^T

Where, Matrix named U is an orthogonal matrix of dimension mxm

$$U = [\, u_1, u_2, .., u_r, u_{r+1}, .., u_m] \tag{2}$$

All the columns of u_i for i = 1, 2, . . . , m, which are known as columns vector form an orthonormal set:

$$u^{\mathrm{T}}{}_i u_j = \delta_{ij} = \begin{cases} 1 \; i = j \\ 0 \; i \neq j \end{cases} \tag{3}$$

And matrix named V is also an orthogonal matrix of dimension nxn

$$V = [\, v_1, v_2, ..., v_r, v_{r+1}, ..., v_m] \tag{4}$$

All the columns of v_i for i = 1, 2, . . . , n, which are known as columns vector form an orthonormal set:

$$v^{\mathrm{T}}{}_i v_j = \delta_{ij} = \begin{cases} 1 \; i = j \\ 0 \; i \neq j \end{cases} \tag{5}$$

Here, S is known as diagonal matrix(all the element are zero except diagonal element) with dimensions m x n and diagonal elements are also known as SV i.e singular values. The matrix S can be showed in following:

$$S = \begin{bmatrix} \sigma_1 & 0 & .. & 0 & .. & 0 & 0 \\ 0 & \sigma_2 & .. & 0 & .. & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & .. & \sigma_r & .. & 0 & 0 \\ 0 & 0 & .. & 0 & .. & \sigma_{r+1} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_n \end{bmatrix}$$

For i running from 1 to n, all the i's are known as Singular Values of the matrix A and it is proved that

$$\sigma_1 \geq \sigma_2 \geq ...\sigma_r > 0, and$$

$$\sigma_{r+1} = \sigma_{r+2} = ...\sigma_N = 0$$

For i running from 1 to n, all the i's are called singular values of the matrix A. All u 's and v 's are known as left and right singular vectors of A [1].

B. Properties of the SVD

Some of the properties of the SVD are mwntioned below:

- The matrix U and matrix V are not unique,however the SVs of the diagonal matrix σ_1,σ_2,...,σ_n are unique.
- Since $A^TA = VS^TSV^T$ so V diagonalizes the matrix A^TA, it follows that the v_j s are eigen vectors of matrix A^TA.
- Since, $AA^T = USS^TU^T$, so matrix U diagonalizes the matrix AA^T and $u_i\acute{s}$ are the eigen vectors of matrix AA^T.
- If r is the rank of matrix A then $v_j, v_j, ..., v_r$ form an orthogonal basis for the range space of matrix A^T, $R(A^T)$, and $u_j, u_j, ..., u_r$ form an orthogonal basis for range space A, R(A).
- The rank of matrix A is equal to the number of its nonzero SVs [4].

IV. Methodology of Singular Value Decomposition applied to Image Processing

A. Singular Value Decomposition Approach for Image Compression

The image data required to present images are reduced with compression of images without losing important features of images. By removing the three fundamental redundancies we can achieve the compression:

1) Redundancy of coding, it represents the information related to an image. The information is stored in a form of code. If a gray level of an image uses more code symbols to store image than necessary to represent each gray scale level than an image can be said as code redundancy.

2) Redundancy of inter-pixel, there are 2 types, spatial and temporal redundancy. Spatial redundancy happens because the neighbouring pixels are correlated and we can predict the values of pixels from the value of neighboring pixels. Pixel carries a small amount of information of an image. To reduce the interpixel redundancy the difference between adjacent pixel can be used to represent an image. And temporal redundancy deal with statical correlation between pixels of different frames of video.

3) Redundancy of psychovisual, it deals with the elimination of information that is not essential for normal human visualization. It does not involve any detailed analysis of image i.e. such as qualitative analysis of every pixel.

One of the property of SVD mentioned before states that number of its nonzero singular values and the rank of matrix A is equal. From many application, rank of matrix increases rapidly when there is decrease in the singular value of the matrix. This propriety can we used for compression by removing the small sigma values and keeping only large and dominating singular values and it is also used in reducing the noise from the image

When we perform SVD image compression, we are selecting only that SVs which has big amount of image data. Using only a few singular values reduces space require to store it in memory and this reduces the size of the image and there are little differences between the compressed and original image which can be seen directly by human visualization. We have shown a detailed process to illustrate the SVD image compression.

$$A = U\Sigma V^T = \sum_{i=1}^{r} \sigma_i u_i v^{\mathrm{T}}{}_i \tag{6}$$

Outer product expansion is a way to represent the matrix A:

$$A = \sigma_1 u_1 {v_1}^{\mathrm{T}} + \sigma_2 u_2 {v_2}^{\mathrm{T}} + ... + \sigma_r u_r {v_r}^{\mathrm{T}} \tag{7}$$

When compressing the image, After SVD decomposition there are 3 matrices in which one is the diagonal matrix(contain only singular value as the diagonal element) and other 2 are orthogonal matrices. Considering the only sum of large singular values and dropping singular values which are small enough. The approximate rank of diagonal matrix k is obtained.

$$A_k = \sigma_1 u_1 {v_1}^{\mathrm{T}} + \sigma_2 u_2 {v_2}^{\mathrm{T}} + ... + \sigma_k u_k {v_k}^{\mathrm{T}} \tag{8}$$

The total storage for A_k will be

$$k(p+q+1) \tag{9}$$

where, p: no of rows in matrix, q: no of columns in matrix.

The value of k which we have chosen is always less than the value of n, and the image formation after doing SVD image compression is still similar to the original image. However, by choosing the different value of k give different images and the storage for it. As a value of k increases the size or memory requirements of images also increases due to increases in pixels of the image. In this project, we have done image compression in python and result of the different value of k are shown in section 4.

B. Measures of Image Compression

The image quality after compression and the factor of compression can be computed if the performance of SVD has to be measured. Formula to calculate the factor of compression i.e compression ratio is given below:

$$C_R = \frac{p * q}{k(p+q+1)} \tag{10}$$

where, p: no of rows in matrix, q: no of columns in matrix.

MSE (Mean Square Error) is one of the measure if the comparision has to be made between the quality of the compressed image and the originall image:

$$MSE = \frac{1}{pq} \sum_{y=1}^{p} \sum_{x=1}^{q} (f_A(a,b) - f_{A_K}(a,b)) \tag{11}$$

where, p: no of rows in matrix, q: no of columns in matrix.

C. Peak Signal Noise Ratio

Maximum value possible for the power of signal divided by the representation of data that is affected by the power of corrupting noise is defined as PSNR (Peak Signal Noise Ratio). In the concept of image compression, Noise represents the error occurs by the compression and the signal represents the actual image data. logarithmin decibel scale is the representation of the same[1,2]:

The PSNR ratio is computed by the PSNR block in the unit of decimbels between the two images. To measure the quality of the compressed image and original image this ratio is used. The quality of the reconstructed image or the compressed image get betters with increase in the ratio.

$$PSNR = 20log_{10}(\frac{MAX_f}{\sqrt{MSE}}) \tag{12}$$

V. Experiments and Results

Colour image is taken for compression. Storage space taken by the original image was 6364.77 kb. The memory space covered by the images has to be reduced, SVD lossy image compression technique is used for that purpose. After that. Image measures considered for this experiment are Mean square error, compression ratio and storage space.

Colour image is converted into grayscale image. Image is decomposed using SVD. Rank of original matrix was roughly 2160. After compression K scaling from 1 to 100, the image was recovered using it's dominant vectors. Below are the results of compression.

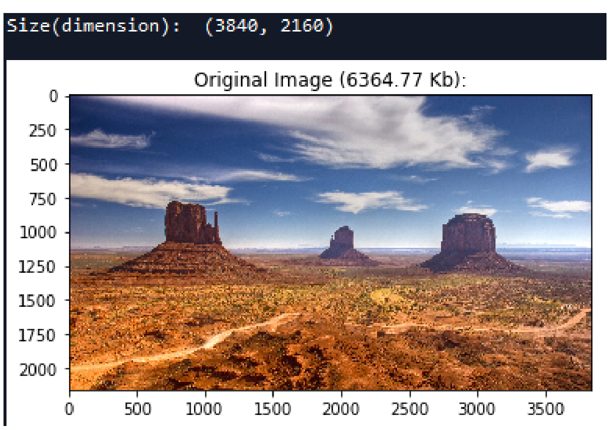

Fig. 3. Original colour image

Above colour image is then transformed to grayscale image using SVD implemented in python, you can find the code in appendix. Then Avd is performed on the grayscale image for compression.

Fig. 4. Grayscale image

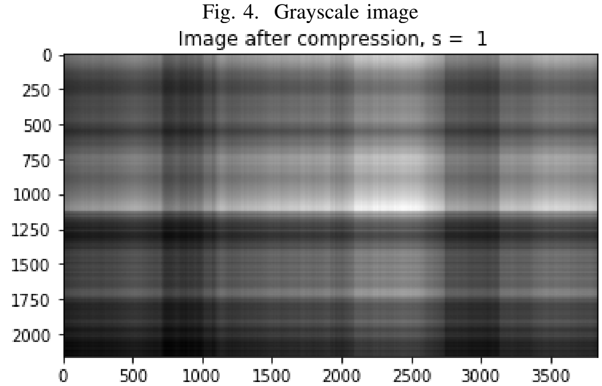

Fig. 5. Compressed image1

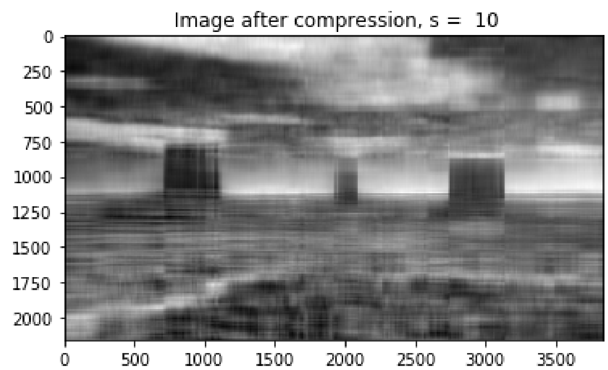

Fig. 6. Compressed image2

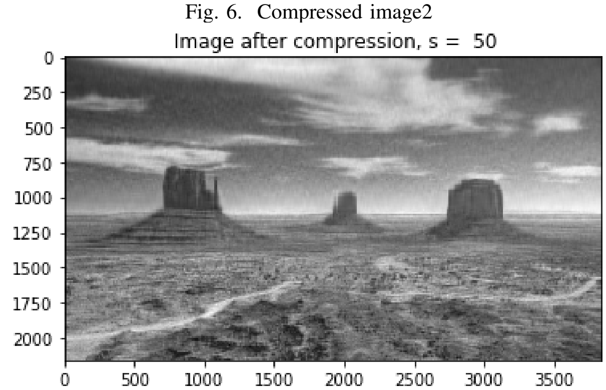

Fig. 7. Compressed image3

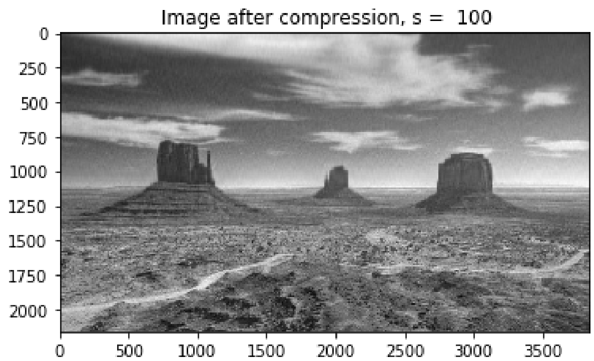

Fig. 8. Compressed image4

TABLE I
RESULTS AFTER COMPRESSION

K (Rank)	Image Measures		
	MSE	***Compression ratio***	***Storage space(kB)***
50	239.1841	27.64	588
60	214.83	23.03	618
70	197.48	19.74	646
80	180.55	17.27	669
90	166.6	15.35	690
100	155.42	13.82	709
300	41.58	4.607	866
500	17.46	2.76	896
Original Image	0	1	6364.77

From the above table we can say that as the rank of matrix increases, the storage space also increases and the compression ratio decreases. Compression ratio tends to 1 as the rank becomes same as original matrix. Here, from the table we take rank 100 with compression ratio 13.82 as the storage space is 709 kB, much lesser than the original image.

VI. Conclusion and Future Events

SVD in the area of linear algebra is applied to this project for digital image processing. Image processing areas are investigated and tested. Original data is split into independent components that are linear and they carries their own information to contribute to the system. SVD can effectively do the same and top of that it is a stable method. We can say that on the basis of the experiments we have performed.

From the computational point of view, SVD is not strong which is the disadvantage of SVD. Due to the excessive work of calculations, its application becomes conditional which is the problem of SVD. On the other hand, compression ratio that is well adapted by the statistical variation in the image is provided by SVD.

The solutions provided by SVD are practical as the image after compression compared to the original one has very satisfactory compression ratio. on the basis of all the results, we can state that SVD is very fast and easy to apply and robust to unusual conditions i.e works good in the enivornment of constrain.

About the aspect of future work. Firstly, the role played by the singular vectors and sigular values can be investigated and studied thoroughly. Secondly, here the work is done in python, it has scope of been executed in other object oriented programming language such as Java, C++ and also MATLAB. Then lastly, One can also work on the very complex images like 3D images, very large 2D imagesor the images with SVD for recognitions and compression.

References

[1] Neethu.K.J, PG scholar, MEA Engineering College, Perinthalmanna, India,sneethuchandram@gmail.com

[2] Sherin Jabbar, Assistant Professor, MEA Engineering College, Perinthalmanna, India, sherinjabbar@gmail.com

[3] Bernd Jahne, "Digital Image Procession", Springer, 2002

[4] Steve J. Leon, Linear Algebra with Applications, Macmillan Publishing Company, New York, 1996

[5] S.G. Kong, J. Heo, B.R. Abidi, J. Paik, and M. A. Abidi. Recent advances in visual and infrared face recognitiona review. Computer Vision and Image Understanding, 97(1):103135, 2005.

[6] Rowayda A. Sadek, Computer Engineering Department, College of Engineering and Technology, Arab Academy for Science Technology & Maritime Transport (AASTMT), Cairo, Egypt

VII. Appendix

Implementation of SVD in python programming language:

```
import os
import numpy as nmp
from PIL import Imge
import matplotlib.pyplot as pl
path = 'desert.jpg'
img = Imge.open(path)
s = float(os.path.getsize(path))/1000
print("Size(dimension): ",img.size)
pl.title("Original Image (
pl.imshow(img) imagegray = img.convert('LA')
imagemat = nmp.array( list(imagegray.getdata(band = 0)), float)
imagemat.shape = (imagegray.size[1], imagegray.size[0])
imagemat = nmp.matrix(imagemat)
pl.figure()
pl.imshow(imagemat, cmap = 'gray')
pl.title("Image after converting it into the Grayscale pattern")
plt.show()
print("After compression: ")
U, S, Vt = nmp.linalg.svd(imagemat)
for i in range(1,300,50):
cmpimg = nmp.matrix(U[:, :i]) * nmp.diag(S[:i]) * nmp.matrix(Vt[:i,:])
pl.imshow(cmpimg, cmap = 'gray')
titleImg = " Image after compression, s =
pl.title(titleImg)
pl.show()
result = Imge.fromarray((cmpimg ).astype(nmp.uint8))
result.save('compressed.jpg')
```