भारतीय प्रौद्योगिकी संस्थान जोधपुर

Indian Institute of Technology Jodhpur

॥ त्वं ज्ञानमयो विज्ञानमयोऽसि ॥

IMAGE COMPRESSION USING PRINCIPAL COMPONENT ANALYSIS (PCA)

Sachin Singh (M24CSE033)
Alok Dutta (M24CSE032)
Jaswant Singh (M24CSE010)

Abstract

This project explores the use of Principal Component Analysis (PCA) for image compression, aiming to reduce image size while maintaining high visual quality. PCA is a dimensionality reduction technique that compresses images by transforming them into a lower-dimensional space, where the most significant features are retained. In this study, PCA is applied separately to the red, green, and blue channels of an image, compressing them by selecting the top principal components and then reconstructing the image. The effectiveness of this method is evaluated by comparing the original and compressed images using metrics like Peak Signal-to-Noise Ratio (PSNR) and compression ratio. The results demonstrate that PCA can significantly reduce image sizes while preserving essential visual details, making it a valuable technique for applications where both image quality and storage efficiency are important. This project also highlights the trade-offs between compression rate and image quality, offering insights into the potential of PCA for image compression in various domains.

Table of Content

1. Introduction 1

2. Theory of Image Compression 2

    2.1. Literature Review 3

    2.2. Principal Component Analysis (PCA) 4

    2.3. PCA for Image Compression 5

3. Application of Image Compression 9

    3.1. Medical Imaging 9

    3.2. Satellite Imagery 9

    3.3. Mobile Storage and Photography 10

    3.4. Streaming and Video Conferencing 10

    3.5. Pattern Recognition and Machine Learning 10

    3.6. Data Storage and Image Retrieval 11

4. Implementation of PCA for Image Compression 12

    4.1. Python Program to compress Image size 12

    4.2. Results 16

5. Conclusion 17

6. References 19

1. Introduction

Today, with the emergence of the digital age, it has increased at a very rapid pace in the demand for storage and transmission of data. Images form an integral part of multimedia data and, hence, require huge storage space and bandwidth during transmission over the internet or on cloud servers. An important application of image compression technique has been the demand for high-resolution imagery by medical imaging, satellite imaging, social media platforms, and e-commerce to reduce file sizes but yet preserve quality [1]. Some popular techniques used for compressing images are JPEG and PNG. JPEG, on the other hand, will reduce its size by letting go of part of the detail of the image that is not as observable, which in the long run will reduce the quality of the image. PNG, on the other hand, is a lossless data compression method that retains the data of the original image, but at the cost of a far much higher storage requirement. These traditional methods are always effectual, even if they do not appear optimal for their intended uses—usually when there is the need to balance quality with extreme compression.

This project looks at exploring a technique for compressing images, namely Principal Component Analysis (PCA), which is commonly used for dimensionality reduction in statistics. Using PCA over images is expected to reduce the sizes of the files of the images, while keeping the important visual information of interest in them intact, giving much smaller sizes of the files of the images after compression and acceptable quality for practical purposes. PCA is a very effective technique in dimensionality reduction for data with high dimensionality, maintaining most of the critical information content within the data. It tries to decompose the original high-dimensional data into a set of mutually orthogonal components, explaining maximum variance in the data. Mostly, images contain a great level of redundancy, which is why the application of PCA in compression is effective. Features that are redundant can be eliminated through the reduction of the dimensionality of image data, while at the same time preserving the intrinsic structure of an image. This contrasts with methods such as the traditional JPEG, which focused on pixel data compression; PCA is a data-driven approach to compression in terms of variance. From this point of view, PCA stands out as a fascinating technique in the image compression phase, because it can achieve the best quality of compression with hardly any loss of information by taking into consideration that most details are basic. This work will point to the effectiveness of PCA in image compression and investigate all possible trade-offs between the rate and quality of compression.

2. Theory of Image Compression

Image compression has become one of the most important technologies in today's digital world, in which the storage, transmission, and retrieval of images all play key roles in such important fields as multimedia, medical imaging, and web technologies. Compression reduces the file size of an image to be stored or transmitted with only negligible effects on visual quality [2]. The main aim in image compression is to diminish the data required by the representation of a certain image while preserving the most vital visual information.

Image compression can be divided into two broad categories: lossless and lossy compression. Lossless compression reduces a file size without any loss in image quality; it means that the image can be rebuilt perfectly from its compressed data. Lossless compression techniques, such as PNG and GIF, are used for images that require high fidelity, such as medical scans or technical drawings. Lossy compression, however, achieves higher compression rates by discarding some of the image data at the expense of quality. JPEG is one of the most common examples of lossy compression. This kind of compression is normally used with photographs and web images where slight reductions in quality can be exchanged for much smaller file sizes.

Efficient image compression is important since many applications require huge volumes of image data to be stored or transmitted efficiently. In many applications, such as Web streaming, social media, satellite imaging, and medical diagnostics reduced-size images, while maintaining visual integrity, have greater transmission rates and require less storage. Whichever image compression technique is used, there has to be a trade-off between compression rate and image quality, since applications differ in their requirements.

In recent years, there has been much research into the use of sophisticated algorithms such as Principal Component Analysis for image compression in regard to dimensionality reduction and retaining the most salient features of an image [4]. Techniques like PCA reduce redundant or less vital information so that images can be compressed well without losing major visual features. This is particularly useful in applications that involve high-quality images with compression, especially in pattern recognition and object detection, among other machine learning-based applications.

2.1. Literature Review

Since such images are usually of a high volume, hyperspectral image compression is very important. JPEG2000 standard is amongst the popular usages that exploit wavelet transforms for efficient compression. However, its performance can be further improved upon with the integration of Principal Component Analysis, pre-processing algorithms for spectral decorrelation, and dimensionality reduction. In a study by Du and Fowler in 2007, PCA was implemented within the framework of JPEG2000 to improve compression efficiency while still retaining all the information needed for anomaly detection. Their results showed PCA-based JPEG2000 outperformed the conventional wavelet-based schemes, in terms of rate-distortion performance, therefore, it may be one of the promising techniques for hyperspectral image compression. [5]

One of the effective techniques being developed for image compression involves the use of Principal Component Analysis in which data can be reduced in dimensions without losing any vital information. One of the successful applications is in medical image compression: PCA focuses on regions of interest, such as the tissue area, and uses simpler models in the background to achieve high compression ratios. This approach keeps the important diagnostic information while drastically reducing the original file size, according to Taur and Tao, 1996; hence, PCA may be considered a potential approach in some specific image compression applications. [6]

A hybrid approach has been suggested for color image compression wherein the redundancy in RGB components of images is reduced by Principal Component Analysis combined with backpropagation learning. It reduces three-dimensional RGB vectors to scalar values for a black and white representation of color images using PCA, while a backpropagation algorithm is used to restore the original image. This was demonstrated by Clausen and Wechsler 2000 that the approach compresses color images effectively while retaining relevant visual features, hence a powerful tool in efficient image storage and transmission.[7]

The rationale behind PCA is that the method minimizes dimensionality by transforming correlated data into uncorrelated components. In application to digital image compression, PCA effectively reduces data redundancy while retaining major image features. This technique transfers huge dimensional image data into a low-dimensional space in which most of the variances of the image are held, hence achieving tremendous data storage reduction with minimal loss of quality. Santo

(2012) applied this technique and proved that it is one of the best methods for digital image compression where much distortion can have a minimal effect.[8]

Block-based Principal Component Analysis: This is a very competent technique in which medical images can be compressed by processing image data in small blocks that are easy to handle. This approach was explored by Lim et al. in 2014 for performing PCA either block-by-block or by first converting blocks into rows before carrying out PCA. Indeed, block-based PCA applied to compress medical images, like digital fundus images, achieves a good compromise that offers a high compression ratio without necessarily losing relevant diagnostic information. This would be very useful in medical applications which are in high demand for efficiency and accuracy.[9]

2.2. Principal Component Analysis (PCA)

PCA is one of the most common statistical methods in data analysis, machine learning, and image-processing studies. The major use of PCA is to reduce the dimensionality in big datasets, retaining a great deal of variability in the data. For this purpose, PCA will transform the data into a new coordinate system where the variance of the data becomes maximized along each axis so that patterns can be identified and complex data simplified.

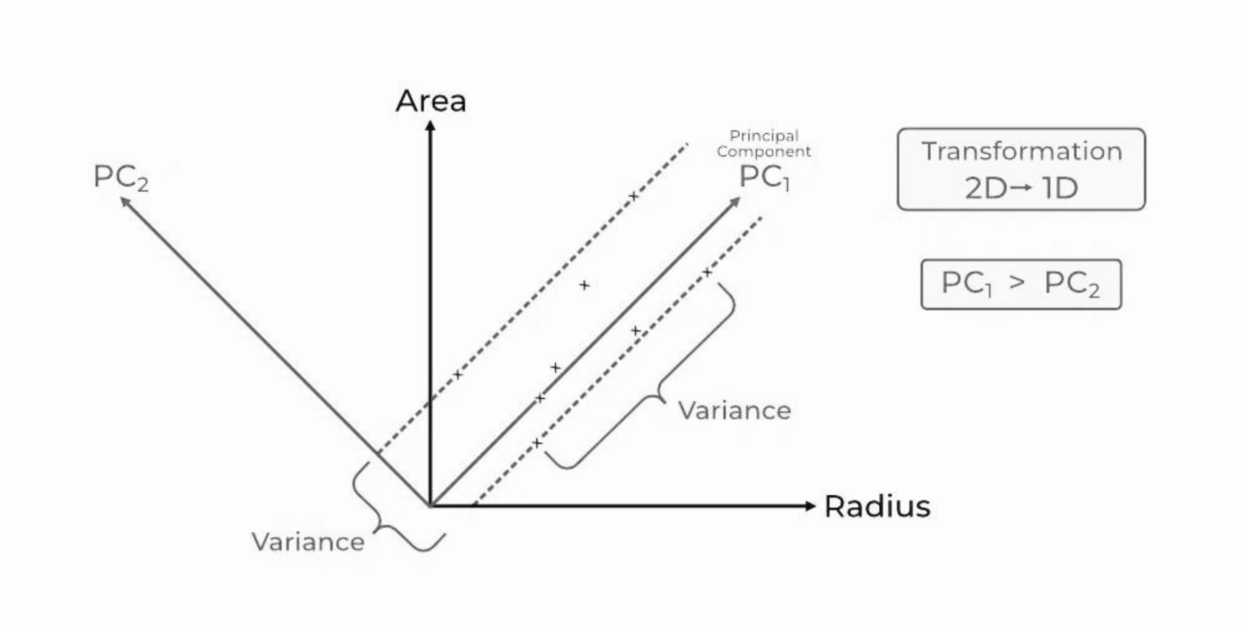


Figure 1 – Principal Component Analysis (PCA)

In essence, PCA identifies the directions, or principal components, along which the data varies the most. These principal components are orthogonal to each other, meaning they are independent, and each one accounts for a portion of the data's variance. The first principal component captures the maximum variance in the data, followed by the second, which captures the next highest variance, and so on. By selecting only the top principal components, PCA enables dimensionality reduction, where the data is projected into a lower-dimensional space, retaining the most critical information while discarding less significant details.

Mathematically, PCA relies on the eigenvalue decomposition of the covariance matrix of data. In this matrix, the eigenvectors correspond to the directions of principal components, while the corresponding eigenvalues indicate how much variance every component explains. By selecting the top eigenvectors associated with the largest eigenvalues, PCA projects the original data onto a lower-dimensional subspace, drastically reducing the number of features necessary to describe the data while retaining most of the original information.[1]

In image processing, the most natural application of PCA is implemented in image compression. Images are naturally high-dimensional data; for example, most images contain thousands or millions of pixels. In images, PCA can be applied to decrease the level of dimensionality by focusing on those key features contributing to the structure of the image visually. The PCA procedure allows one to compress an image using only a subset of the principal components in a particular manner, allowing for gross reduction in file size and retaining most of the details important for a viewer within the image. It is a powerful technique in applications requiring both high compression rates and the preservation of image quality.

In this project, PCA is used to compress images by reducing each respective color channel's dimensionality—separately for red, green, and blue. Retaining only the most important components in each channel results in small loss of visual quality when the image is reconstructed and supports the fact that PCA can be very effective in performing image compression.

2.3. PCA for Image Compression

One of the effective methods for image compression is Principal Component Analysis, whereby the aim is to reduce the size of the image while preserving the important features within the image. Images are normally of a high-dimensional nature, with every pixel depicting a feature. Thus, PCA can be used for dimensionality reduction by identifying and retaining only the most key features. We can reduce the storage requirements and transmission time by compressing the image through PCA, and for that matter, it is very useful in several applications such as in multimedia, medical imaging, and pattern recognition.

The process of using PCA for image compression can be broken down into the following key steps[10]:

i. Convert Image to Grayscale

For simplicity, most of the image compression starts by converting an image to grayscale. The grayscale image can be viewed as a 2D matrix where each element would correspond to the intensity value of a pixel going from 0 to 255. Wherever color images are used, treating each color channel separately (Red, Green, Blue) and applying PCA individually to each channel is being done.

ii. Center the Data

Next comes the centering of the pixel values by subtracting the mean of each pixel across the whole image. This will make zero mean data, which makes applying PCA easier. Mathematically speaking, given an image represented as matrix X, the centered data $X_{centered}$ is computed as:

$$X_{centered} = X - \mu$$

Here μ is the mean of each pixel.

iii. Calculate the Covariance Matrix

The covariance matrix is calculated to capture the relationships between different pixels in the image. This matrix is essential for understanding how the intensity of one pixel varies with another and forms the foundation for identifying the principal components, which capture the most variance in the image.

The covariance matrix C is calculated as:

$$C = \frac{1}{m-1} X_{centered}^{T} X_{centered}$$

Here m is the number of pixels, and $X_{centered}^{T}$ is the transpose of the centered image matrix.

iv. Compute Eigenvalues and Eigenvectors

Then, one computes its eigenvalues and eigenvectors, which can be explained through the directions of maximum variances in an image plus how much variance each of those eigenvectors captures. In PCA-based image compression, we only keep the first k eigenvectors (principal components) with the highest eigenvalues that contain the most important visual information, thus lowering the dimension of the data.

v. Project the Image onto the Principal Components

After finding the top principal components, the original image data is projected onto this reduced subspace. The projection of the centered data onto the eigenvectors W is calculated as:

$$Z = X_{centered}W$$

where Z is the compressed representation of the image, and W is the matrix of selected eigenvectors.

This step effectively reduces the dimensionality of the image by transforming it into a smaller set of features that capture most of the visual variance.

vi. Reconstruct the Image

To reconstruct the compressed image, the projection is reversed by multiplying the compressed data Z with the selected eigenvectors and adding back the mean that was subtracted earlier:

$$X_{reconstructed} = ZW^{T} + \mu$$

This reconstructed image approximates the original, but with fewer dimensions and a smaller file size.

vii. Compression Efficiency

The number of principal components k directly affects the compression ratio and the quality of the reconstructed image. Choosing fewer components results in higher compression but also a greater loss of image detail, while selecting more components retains more detail at the cost of larger file size.

The compressed size can be quantified by the number of principal components retained and the size of the eigenvector matrix, while the effectiveness of the compression can be measured using metrics like Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE) between the original and reconstructed images.

viii. Advantages and Trade-offs

PCA-based image compression has several advantages:

- It effectively reduces file size while preserving critical visual features.
- It is computationally efficient for smaller images or moderate compression levels.
- It provides a tunable balance between compression rate and image quality by adjusting the number of retained principal components.

However, PCA also has some limitations:

- It is most effective for images with correlated pixel values, which might not be the case in all images.
- For highly compressed images (i.e., when a few principal components are retained), the quality can degrade significantly, leading to blurred or distorted images.
- The PCA is a linear method and does not capture any strongly complex, non-linear features in an image as well as other compression techniques.

Therefore, PCA supplies an easy yet powerful method of compressing images by decreasing the amount of data required to describe an image without a significant loss in visual quality. By focusing on the most variance within the data, PCA optimally reduces file sizes and is highly applicable in areas where large volumes of image data need to be stored and transmitted.

3. Application of Image Compression

Indeed, Image Compression is an essential tool for the management of large amounts of image data created in many industries today. In essence, it makes images smaller in size without much loss in the quality, thereby enhancing the storage efficiency and allowing quicker transmission and reducing the computational burden involved in processing during tasks involving data. This is an extremely significant requirement in applications such as medical imaging, satellite imagery, or mobile storage where high-resolution images are constantly being produced for storage, transmission, and processing.

3.1. Medical Imaging

Large high-resolution images created by MRI, CT scans and X-rays to be stored for later use and widely used in large-size image communications over networks for diagnosis or consultation [11]. The size of these large images can be a storage and communication burden, especially in telemedicine applications. Image compression reduces the file size without losing necessary diagnostic information. In such a situation, lossless compression is preferred and avoids the loss of information during compression thereby ensuring that the medical images preserved retain all the information. This compression-based method can be done by applying PCA-based compression to reduce the sizes of images. Information does not get lost during the storage and transmission of data.

3.2. Satellite Imagery

Some applications of the given set of satellite images include weather forecasting, monitoring of environmental formations, and defense. All these satellite images are taken at very high resolutions, meaning that those large datasets come as a challenge in storage and transmission of those in raw format. Applying the image compression techniques reduces the sizes of the satellite images much more efficiently and facilitate storage and transmission. In such a scenario, this minor degradation in image quality often turns out to be pretty fine because it may not influence the general usage of that image [12]. PCA-based compression is very helpful in compressing the spectral bands of the satellite images; PCA gives a way to do data reduction without compromising major information.

3.3. Mobile Storage and Photography

As smartphone cameras continue to grow in resolution, a lot of consumers find themselves storing thousands of photos on the devices. Compression of images relieves the load on device storage without seriously compromising quality [13]. Furthermore, most social media and cloud storage services compress images, so they upload and download quickly over cellular bandwidths. Here, one can apply PCA-based compression with good visual fidelity so that the sizes of the images can be reduced, and thus in applications which have storage as a limiting factor, one may balance between quality and space for the users.

3.4. Streaming and Video Conferencing

This is especially important in streaming websites for videos and video conferencing where images and frames of the video are continuously transmitted over the internet[14]. For interference-free communication, such websites have to compress images appropriately not compromising on degradation at the cost of increased bandwidth requirements. Technologically, lossy compression techniques like PCA-based compression can indeed achieve this by compressing every frame of the video or image stream in such a manner so as to allow more data to be transmitted over a channel with acceptable quality at higher speeds.

3.5. Pattern Recognition and Machine Learning

PCA is used in applications dealing with huge databases of images, such as for facial recognition or pattern recognition and object detection. These images carry redundant information; they take a long time to train and usually add up to a high storage cost[15]. Reducing the dimensionality of these images by finding out and then retaining the important features that exist in them will provide an efficient process for processing and managing data. PCA compression reduces the data, which results in reducing its complexity. This helps speed up the model training and keeps important information for accuracy in predictions.

3.6. Data Storage and Image Retrieval

This becomes important at large scales, thinking of applications like image databases in digital libraries, e-commerce sites, or scientific repositories. Compression of images reduces disk space used-and-of course, retrieval and display speed [16]. PCA-based compression might ease archiving and retrieval of images if it reduces the number of data required to be stored to represent high-quality images. For example, PCA can reduce the dimensionality of images stored for any image retrieval system so that algorithms used for searching become efficient and fast retrieval of images from the database.

4. Implementation of PCA for Image Compression

This compression project used Python for doing Principal Component Analysis on images. The implementation supplied here reduced the dimensionality of the image without much compromise on its visual quality. The implementation applies PCA separately to each of the RGB color channels and reconstructs the compressed image by retaining only the most significant components from every color channel. Step-by-step explanation of how PCA is applied in compressing an image of the project is as follows:

4.1. Python Program to compress Image size

Step 1: Loading the Image

The process starts with the loading process of the image to a numerical form which allows making some numerical operations. Using a library - imageio, an image was loaded and further on converted to a kind of NumPy array. Then the image was separated into the individual RGB channels.

```
import imageio.v2 as imageio
import numpy as np
import matplotlib.pyplot as plt

# Load the image
a = imageio.imread("/content/sample_data/dhoni.jpg")

# Convert the image to a numpy array and extract RGB channels
a_np = np.array(a)
a_r, a_g, a_b = a_np[:, :, 0], a_np[:, :, 1], a_np[:, :, 2]
```

This step resamples the image with ultimate capability to capture each color channel so that PCA is applied on each one separately. It is because the case for each channel being full of different intensity values, thus it is assured that by applying PCA to each of the individual channels color information is kept during compression.

Step 2: PCA-Based Compression

This function pca_compression() applies PCA-based compression. The scheme begins by centering the data through subtracting the mean of the pixel value by channel, applying SVD Singular Value Decomposition, or on the centered data matrix that decomposes the latter into principal components; then it retains only a few of the top components. This number of retained top components can be controlled using numpc. This compresses an image representation which originally had a high-dimensional representation. Finally, it is reconstructed using the selected principal components.

```
# Function to perform PCA-based compression
def pca_compression(image_2d, numpc=100):
    # Center the data (subtract mean)
    mean = np.mean(image_2d, axis=0)
    centered_data = image_2d - mean

    # Perform Singular Value Decomposition (SVD)
    U, S, Vt = np.linalg.svd(centered_data, full_matrices=False)

    # Select the top numpc components
    eigenvectors_reduced = Vt[:numpc, :]

    # Project the data onto the reduced eigenvector space
    compressed_data = np.dot(centered_data, eigenvectors_reduced.T)

    # Reconstruct the image from the compressed data
    reconstructed_data = np.dot(compressed_data, eigenvectors_reduced) + mean

    # Calculate the compressed size (numpc components * width of the image)
    compressed_size = compressed_data.size + eigenvectors_reduced.size

    return np.uint8(np.clip(reconstructed_data, 0, 255)), compressed_size
```

In this implementation:

- The SVD function decomposes the image into its components, where only the top numpc components are retained, thereby reducing the dimensionality of the image.

- The reduced data is multiplied with the chosen eigenvectors and the mean is added to recover the original scale.

- The final reconstructed image is clipped onto a valid range of 0-255 so that it becomes also visually valid.

Step 3. Reconstructing the Compressed Image

Then all the compressed and reconstructed color channels are combined to finally compress the image. In this stage, the color channels that have been compressed separately are then combined into one image.

```
# Perform PCA-based compression and reconstruction for each channel
reconstructed_channels = []
compressed_sizes = []
for channel in [a_r, a_g, a_b]:
    recon_channel, compressed_size = pca_compression(channel, numpc=100)
    reconstructed_channels.append(recon_channel)
    compressed_sizes.append(compressed_size)

# Combine reconstructed channels into a color image
recon_color_img = np.dstack(reconstructed_channels)
```

Thus, the compressed color image can be assembled from the compressed color channels: Red, Green, and Blue. Each of them has separately been compressed followed by reconstruction using PCA; this way, nearly all of the original visual information of the image is preserved even though the file size has reduced drastically.

Step 4: Displaying the Original and Compressed Image

To visually compare the original and compressed images, the matplotlib library is used to display both versions of the image. This help for a fast visual representation of the image quality before and after compression of image.

```
# Display original image
plt.imshow(a_np)
plt.title("Original Image")
plt.axis('off')
plt.show()

# Display the reconstructed (compressed) image
plt.imshow(recon_color_img)
plt.title("Reconstructed (Compressed) Image")
plt.axis('off')
plt.show()
```

This step displays the original image and the PCA-compressed image side by side. Even though the compressed image uses fewer principal components to represent the original data, it is still visually similar, demonstrating the effectiveness of PCA for image compression.

Step 5: Calculating Image Size and Compression Ratio

One of the key goals of image compression is to reduce the file size. The original image size and the compressed image size are calculated and printed for comparison. This helps in numerical analizing the compression achieved through PCA.

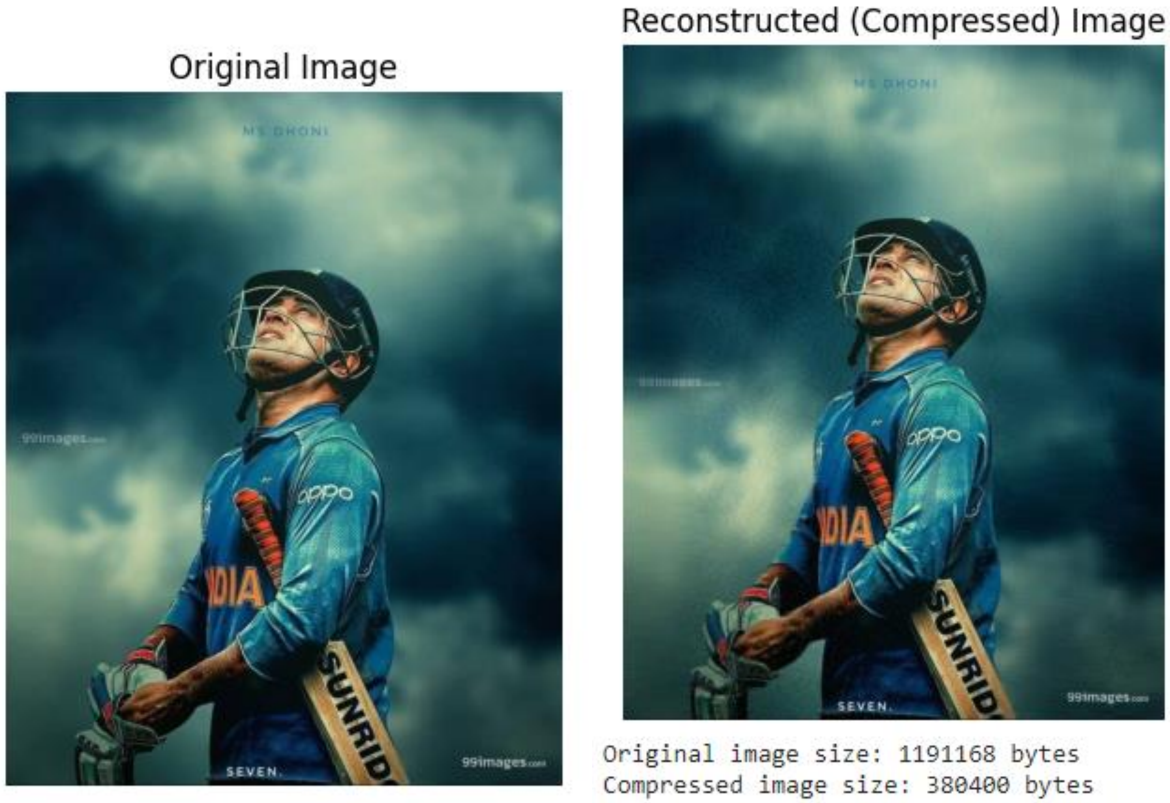
```
from sklearn.metrics import mean_squared_error

# Function to calculate PSNR
def calculate_psnr(original, compressed):
    mse = mean_squared_error(original.flatten(), compressed.flatten())
    if mse == 0:
        return float('inf')  # PSNR is infinite if MSE is zero
    pixel_max = 255.0
    psnr = 20 * np.log10(pixel_max / np.sqrt(mse))
    return psnr
```

The PSNR returned is between the original and the reconstructed images. The higher the value of PSNR, the closer in quality to that of the original image the compressed image is, hence the better the compressor algorithm is.

This is the example of image compression using PCA: it dramatically reduces dimensionality of the image preserving all its appreciable features visually. If PCA is independently applied to each color channel and then reconstructs the image then the file size of the latter becomes much more smaller, keeping the quality of the image the same. The results show that PCA-based compression technique finds an intermediate balance between the reduction of storage space and the preservation of image quality, so it is a useful technique for applications such as mobile storage, satellite imagery, and pattern recognition.

4.2. Results

In the process of applying PCA on images, compression was possible while still having excellent quality images. The figures for the compressed image are nearly original clarity even if it has been reduced down to fewer principal components. The reduction of sizes of the original images is very vivid and further manifest the strength of PCA. PSNR values for the compressed image demonstrate that the quality is basically retained during compression. Follow are screenshots of the original and compressed images:



```
Original image size: 1191168 bytes
Compressed image size: 380400 bytes
```

Figure 2: Result of Image compression using PCA

After compressions the original and the compressed images are compared. The original has 1,191,168 bytes, while the compressed version will have only 380,400 bytes. Of course, it does answer the long-standing question of nearly a 68% reduction of the size of the file. Still, the difference is minimal, particularly in the visual perspective between the two images, which epitomizes that PCA will not fail to retain all the essential visual features even after compression. This emphasizes the functionality of the technique in real life where storage and transmission efficiency are of high importance.

5. Conclusion

Here, in this project, have illustrated how PCA can be utilized for the purpose of image compression-an efficient technique with which one can decrease the number of features in big data while retaining most of the information. This is a project meant to show that it is very much possible to compress a given image with side effects taken in terms of loss in storage size and subsequently its quality. As it will be made possible through detailed exploration of the theoretical foundation, practical application, and implementation within Python, actually, it is a report that brings to light strength and trade-offs in PCA-based image compression.

We began with the description of why image compression is important in a multitude of real-life applications, like medical imaging, satellite imagery, and mobile storage. For all these fields, image compression reduces the storage needs, supports faster transfer of data, and decreases costly computations linked with processing large numbers of image files. The paper made emphasis on how PCA-based image compression can be applied to interesting domains like pattern recognition, machine learning, and image retrieval, which gain reduced dimensionality of data representation at the expense of better performance without data loss.

The theory paper explaining the mathematical base from which PCA operates stated how PCA can take data and somehow translate it into a new coordinate system in which, so to speak, variance is maximised along each principal component. Performing PCA involves eigenvalue decomposition of the covariance matrix: finding the axes to which the data actually varies the most and thus reducing its dimensionality by selecting only the first few principal components. It especially applies to image compression because of dimension reduction, that opens up much space without compromising much on the quality of the image.

The main focus of the report has been on how PCA can be used for compressing images. Toward this end, we explained in detail how PCA was so far used in compressing an image by working with the color channel of images, that are Red, Green, and Blue. By using the SVD, the top principal components were selected in representing the majority of the features of the image. We can reconstruct an image that is compressed down to the extreme, one that looks very much like the original, but at much smaller size by keeping some subset of these components. But what was impressive about the implementation was that each channel may be compressed independently and then combines into one final color image.

To assess the results visually, the original images were compared with the compressed images, which showed a few differences in quality. Measuring quality was done using PSNR as a measure of the quality with which the original image is compressed; high PSNR means that an image, even when dimensionally reduced, retained a high visual fidelity. The report further quantified the compression achieved by comparing original image size with that of the size of the compressed version and demonstrated real advantages in store and transmission requirements achieved by PCA.

We also studied the trade-offs between compression and image quality. By adjusting the number of principal components that were retained during compression, a balance is struck between minimizing file size and maintaining acceptable image quality. The report discussed how this would indeed reduce the number of components the image retains but allows for a higher compression ratio, which also introduces the loss in detail from the compressed image. But retaining more features improved the image quality but degraded the compression performance. That was a nice trade-off important to applying PCA to many settings in which one wants more storage efficiency than would be tolerated without more lost data than allowed.

In general, this project has well demonstrated the effectiveness of PCA in the compression of images. PCA is effective at reducing file sizes of images and saving much of the important visual information; hence, for a number of applications where storage efficiency is as important as data transmission speed. Results for this method show applications that are very important in areas where, for example, more data in images must be efficiently processed like medical images in satellite images and in mobile storage. Although it introduces a trade-off between compression and quality PCA can therefore preserve the most important features of an image, thus making it very relevant in image compression tasks. The PCA method provides a very versatile and efficient way of dealing with large-scale image data in very diverse fields.

6. References

[1] A. Maćkiewicz and W. Ratajczak, "Principal components analysis (PCA)," *Comput Geosci*, vol. 19, no. 3, pp. 303–342, Mar. 1993, doi: 10.1016/0098-3004(93)90090-R.

[2] A. Subramanya, "Image compression technique," *IEEE Potentials*, vol. 20, no. 1, pp. 19–23, 2001, doi: 10.1109/45.913206.

[3] A. J. Hussain, A. Al-Fayadh, and N. Radi, "Image compression techniques: A survey in lossless and lossy algorithms," *Neurocomputing*, vol. 300, pp. 44–69, Jul. 2018, doi: 10.1016/J.NEUCOM.2018.02.094.

[4] M. Mudrová and A. Procházka, "PRINCIPAL COMPONENT ANALYSIS IN IMAGE PROCESSING."

[5] Q. Du and J. E. Fowler, "Hyperspectral image compression using JPEG2000 and principal component analysis," *IEEE Geoscience and Remote Sensing Letters*, vol. 4, no. 2, pp. 201–205, Apr. 2007, doi: 10.1109/LGRS.2006.888109.

[6] M. Mofarreh-Bonab and M. Mofarreh-Bonab, "A New Technique for Image Compression Using PCA."

[7] C. Clausen and H. Wechsler, "Color image compression using PCA and backpropagation learning," *Pattern Recognit*, vol. 33, no. 9, pp. 1555–1560, Sep. 2000, doi: 10.1016/S0031-3203(99)00126-0.

[8] R. Do and E. Santo, "Principal Component Analysis applied to digital image compression Utilização da Análise de Componentes Principais na compressão de imagens digitais," 2012.

[9] H. Asyrani. Bin Sulaiman, *2014 I4CT : 1st International Conference on Computer, Communications, and Control Technology : proceedings : Fave Hotel, Langkawi, Kedah, Malaysia, 2-4 Sept 2014*. IEEE, 2014.

[10] S. C. Ng, "Principal component analysis to reduce dimension on digital image," *Procedia Comput Sci*, vol. 111, pp. 113–119, Jan. 2017, doi: 10.1016/J.PROCS.2017.06.017.

[11] D. A. Koff and H. Shulman, "An Overview of Digital Compression of Medical Images: Can We Use Lossy Image Compression in Radiology?"

[12] K. S. Gunasheela and H. S. Prasantha, "Satellite Image Compression-Detailed Survey of the Algorithms," *Lecture Notes in Networks and Systems*, vol. 14, pp. 187–198, 2018, doi: 10.1007/978-981-10-5146-3_18.

[13] A. N. Skodras, "The Jpeg2000 Image Compression Standard in Mobile Health," *Mhealth*, pp. 313–327, Jan. 2006, doi: 10.1007/0-387-26559-7_23.

[14] S. Katsigiannis, D. Maroulis, and G. Papaioannou, *A GPU based real-time video compression method for video conferencing*.

[15] L. Liu, Y. Wang, and W. Chi, “Image Recognition Technology Based on Machine Learning,” *IEEE Access*, 2021, doi: 10.1109/ACCESS.2020.3021590.

[16] A. S. Abdelrahim, M. A. Abdelrahman, A. Mahmoud, and A. A. Farag, “Image retrieval based on content and image compression,” *2011 International Conference on Multimedia Technology, ICMT 2011*, pp. 6696–6703, 2011, doi: 10.1109/ICMT.2011.6002070.

Appendix

Google Colab Link:

https://colab.research.google.com/drive/1IvO6UP05K-K1DCQgUGqybF0AHVQ7eJ9f?usp=sharing