(Established under Karnataka Act No. 16 of 2013)

UE21MA141B- LINEAR ALGEBRA AND ITS APPLICATIONS

LAA- PROJECT

Session: Jan-May 2023

Branch : CSE

Semester & Section : 4th sem, Section: C

Sl No.	Name of the Student	SRN	Marks Allotted (Out of 5)
1.	Chandrachud Sarath	PES1UG21CS150	
2.	Darsh Agarwal	PES1UG21CS166	
3.	Dhruv Nilkund	PES1UG21CS178	

Name of the Course Instructor :

Signature of the Course Instructor (with Date) : _____

Image Segmentation using EigenVectors and EigenValues (Special Clustering Algorithm)

Introduction

Image segmentation is an important task in computer vision that involves dividing an image into distinct regions or objects based on certain characteristics. One approach to image segmentation is to use eigenvalues and eigenvectors, which can provide a powerful tool for analyzing the texture properties of an image. In this report, we will discuss the mathematics behind this approach and provide a step-by-step procedure for implementing it in MATLAB.

Special Clustering Algorithm

The algorithm for spectral clustering can be expressed mathematically using the following equations:

Step 1: Construct the Graph

Let G = (V, E) be a graph representing the image, where V is the set of vertices and E is the set of edges.

The weight of an edge between vertices i and j is given by w(i, j), which represents the similarity between the pixel values at positions i and j.

Step 2: Compute the Laplacian Matrix

Let D be the diagonal degree matrix, where $D(i, i) = \sum w(i, j)$ for all $j \in V$.

Let A be the adjacency matrix, where $A(i, j) = w(i, j)$ for all $(i, j) \in E$.

The Laplacian matrix L is defined as L = D - A.

Step 3: Compute the Eigenvectors and Eigenvalues

Let λ1, λ2, ..., λn be the eigenvalues of the Laplacian matrix L, sorted in increasing order.

Let v1, v2, ..., vn be the corresponding eigenvectors, where vi is the eigenvector corresponding to eigenvalue λi.

Step 4: Apply Spectral Clustering

Let X be a matrix whose rows are the eigenvectors v2, v3, ..., vk+1, where k is the number of segments desired.

Perform K-means clustering on the rows of X to obtain k clusters, representing the image segments.

Step 5: Post-processing

Apply post-processing techniques such as morphological operations or Gaussian filtering to refine the segmentation boundaries and remove noise.

In conclusion, spectral clustering is a powerful technique for image segmentation that combines the principles of graph theory and linear algebra. Its effectiveness has been demonstrated in various applications, and it remains an important tool in the field of computer vision and image processing.

CODE :

```
img = imread('input.jpg');

img = rgb2gray(img);

% Construct similarity graph

epsilon = 100; % control the weight of edges

[height, width] = size(img);

n = height * width;

W = zeros(n, n);

for i = 1:n

    for j = i+1:n

        diff = double(img(i)) - double(img(j));

        w = exp(-(diff^2) / epsilon);

        W(i, j) = w;

        W(j, i) = w;

    end

end

% Compute Laplacian matrix

D = diag(sum(W, 2));

L = D - W;

% Compute eigenvectors and eigenvalues
```

```
[eigenvectors, eigenvalues] = eig(L);

[sorted_eigenvalues, sorted_indices] = sort(diag(eigenvalues));

sorted_eigenvectors = eigenvectors(:, sorted_indices);

% Perform K-means clustering on eigenvectors

k = 3; % number of segments

X = sorted_eigenvectors(:, 2:k+1);

[idx, centers] = kmeans(X, k);

% Reshape segmentation

segmented_img = reshape(idx, [height, width]);

% Display results

figure;

subplot(1,2,1);

imshow(img);

title('Original Image');

subplot(1,2,2);

imshow(segmented_img, []);

title(sprintf('Segmented Image (%d segments)', k));
```

SVD For Image Compression

Introduction:

Image compression is a technique used for reducing the size of data used to represent digital images. The goal is to represent image files with the lowest number of bits possible. Redundancies present in the image file are exploited in order to reduce the number of bits. SVD-based image compression is a popular technique that uses the singular values decomposition of an image matrix to reduce its storage size. In this report, we will provide an in-depth overview of SVD-based image compression, discuss its mathematical foundations, present some examples of its applications, and evaluate its performance.

Overview of SVD-Based Image Compression:

SVD-based image compression works by decomposing an image matrix into its singular values and eigenvectors. The singular values represent the importance of each eigenvector in the matrix, and they can be sorted in descending order to determine which ones to keep and which ones to discard. The eigenvectors are the directions in which the image varies the most, and they can be used to reconstruct the original image with some loss of information.

To apply SVD-based compression, the image matrix is first decomposed into three matrices: U, Σ, and V. The matrix U contains the left singular vectors, the matrix Σ contains the singular values in diagonal form, and the matrix V contains the right singular vectors. The singular values in Σ are sorted in descending order, and the corresponding singular vectors in U and V are rearranged accordingly. Then, a specified number of singular values are kept and the rest are discarded. The compressed image can then be reconstructed by multiplying the truncated U, Σ, and V matrices.

The number of singular values to keep determines the level of compression and the quality of the resulting image. A higher number of singular values will result in a better-quality image but a smaller compression ratio, while a lower number of singular values will result in a higher compression ratio but a lower-quality image.

Mathematical Foundations:

The mathematical foundations of SVD-based image compression lie in linear algebra and matrix theory. The SVD theorem states that any matrix can be decomposed into the product of three matrices: U, Σ, and V. The matrix Σ contains the singular values, which are positive and real, and the matrices U and V contain the corresponding singular vectors.

In the context of image compression, the matrix represents the image data, and the singular values and vectors represent the spatial frequencies and patterns in the image. By discarding the least important singular values and vectors, it is possible to compress the image while retaining its essential features.

Applications of SVD-Based Image Compression:

SVD-based image compression has many applications in fields such as satellite imaging, medical imaging, and video compression. It is also used in consumer electronics, such as digital cameras and mobile devices, where storage space is limited. One notable application of SVD-based image compression is in the field of remote sensing. Satellite images are typically very large and require significant storage space. SVD-based compression can be used to reduce the size of these images while preserving their quality, making them easier to transmit and store. Another application of SVD-based compression is in medical imaging, where large amounts of data need to be analyzed and stored efficiently. SVD-based compression can be used to reduce the size of medical images without sacrificing their diagnostic value.

Advantages and Disadvantages of SVD-Based Image Compression

SVD-based image compression has several advantages over other compression techniques. It can achieve high compression ratios while preserving a high level of image quality. Finally, it is computationally efficient, making it suitable for use in real-time applications.

However, SVD-based image compression also has some disadvantages. One of the main disadvantages is that it requires a large amount of memory to store the singular values and eigenvectors. This can be a problem for large images or when processing multiple images at once. Additionally, the level of compression achieved with SVD-based compression depends on the number of singular values and eigenvectors kept, and choosing the optimal number can be a difficult task.

Code:

```
inImage=imread('test1.jpg');

inImage=rgb2gray(inImage);

inImageD=double(inImage);


% decomposing the image using singular value decomposition

[U,S,V]=svd(inImageD);


% Using different number of singular values (diagonal of S) to compress and

% reconstruct the image

dispEr = [];

numSVals = [];

for N=5:25:300

        % store the singular values in a temporary var
```

```
        C = S;


        % discard the diagonal values not required for compression
    C(N+1:end,:)=0;
    C(:,N+1:end)=0;


        % Construct an Image using the selected singular values
        D=U*C*V';



        % display
        figure;
        buffer = sprintf('Image output using %d singular values', N)
    imshow(uint8(D));
    title(buffer);
end
```

Original

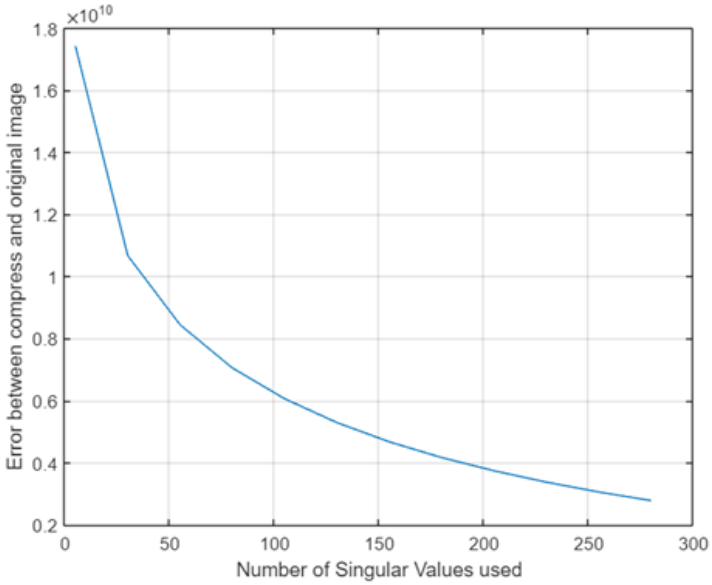
Image output using 30 singular values

Image output using 80 singular values

Image output using 130 singular values

Image output using 230 singular values


Image output using 280 singular values


×10^10
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
Error between compress and original image
0
50
100
150
200
250
300
Number of Singular Values used

Conclusion:

SVD-based image compression is a powerful technique for reducing the storage requirements of image data while preserving a high level of image quality. It has many applications in fields such as satellite imaging, digital cameras, and medical imaging. However, it requires careful selection of the number of singular values and eigenvectors to keep in order to balance the competing goals of compression and image quality. Overall, SVD-based image compression is a valuable tool for anyone working with large amounts of image data.

DEEP FAKE DETECTION- Mathematical concepts behind the AI Concept

INTRODUCTION

Deep-fakes are a form of artificial intelligence-based image or video manipulation that involves creating convincing, but fake, images or videos of people saying or doing things they did not actually say or do.

Deep-fakes have become increasingly sophisticated in recent years, with the ability to manipulate facial expressions, body movements, and even voice to create realistic videos that are hard to distinguish from the real thing.

As deep-fakes become more sophisticated and easier to create, it is important to have effective tools for detecting and preventing them. This has led to a growing field of research and development in deep-fake detection, which involves using various techniques from computer vision, machine learning, and deep learning to identify and distinguish between real and fake images or videos.

APPLICATION OF LINEAR ALGEBRA IN DEEP FAKE MODELS

Like every Artificial intelligence model, the basis of computation is mathematical concepts. In the deep field of deep-fakes, we are primarily dealing with media such as video, and still images. Therefore, the optimal approach is to represent these media as matrices of pixels.

To operate on these matrices and achieve our desired output, we will be implementing various matrix manipulation and *linear algebra*.

Principal Component Analysis (PCA)

Principal Component Analysis is a linear algebra technique used to identify patterns and correlations in high-dimensional data.

In deep-fake detection, PCA can be used to identify the features that are most important for distinguishing between real and fake images. By reducing the dimensionality of the image data using PCA, it is possible to identify the features that are most informative for detecting deep-fakes.PCA identifies the principal components that capture the most variation in the data. By selecting the top principal components, we can reduce the dimensionality of the data while still retaining the most important information.

Singular value decomposition (SVD) [Refer Page:5 for detailed explanation]

Singular value decomposition in the case of deep-fake detection, we can use SVD to extract the most important features from an image or video and use them to distinguish between real and fake content. We can apply SVD to a set of facial images or videos and extract the most important facial features, such as eye position, mouth shape, and facial expression. We can then use these features to train a machine learning model to classify new images or videos as real or fake.

We can apply SVD to the matrix of flattened images. SVD decomposes the matrix into three components: a left singular matrix, a diagonal matrix, and a right singular matrix. The diagonal matrix contains the singular values, which represent the amount of variation in the data that is explained by each principal component.

Video deep-fake classification using CNN

Convolutional Neural Networks are a type of deep learning model that are widely used for image recognition and classification tasks. In deep-fake detection, CNNs can be used to identify the features that are most important for distinguishing between real and fake images. By training a CNN on a large dataset of real and fake images, it is possible to learn the features that are most indicative of deepfake manipulation.Each convolutional layer can be represented as a matrix multiplication between the input image and a set of learnable filters, followed by a nonlinear activation function such as ReLU. The output of the layer is then passed to the next layer for further processing. The technique is valid for images, but preferred for videos.

A very competent CNN model for such purposes is VGG (Visual geometry group).

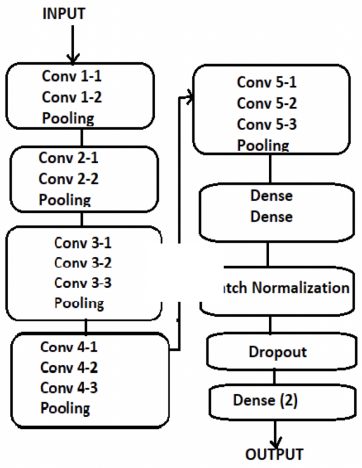


Figure Architecture of the fine-tuned VGG model used

For example, let's say we have a CNN model that takes in an image and outputs a probability of it being real or fake. The input image is transformed through a series of convolutional and activation layers, resulting in a low-dimensional representation of the image. The weights in each layer are learned through backpropagation, where gradients are calculated using matrix operations such as matrix multiplication and element-wise multiplication.

$$f(t) * g(t) := \underbrace{\int_{-\infty}^{\infty} f(\tau) g(t-\tau)\, d\tau}_{(f*g)(t)},$$

Image deep-fake classification using GAN

Generative Adversarial Networks (GANs) are a type of deep learning model that are widely used for image generation and manipulation tasks. In deep-fake detection, GANs can be used to generate fake images that mimic the characteristics of real images. By comparing the generated images to real images, it is possible to identify the features that are most indicative of deepfake manipulation.

They consist of two neural networks: a generator and a discriminator. The generator network is trained to generate images that resemble the real images in the dataset, while the discriminator network is trained to distinguish between the real and fake images.

From a linear algebra perspective, GANs can be viewed as a game between the generator and the discriminator, where the generator tries to minimize the loss function of the discriminator by generating realistic images, while the discriminator tries to maximize the loss function by correctly classifying the real and fake images.

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$

References

Image Segmentation

Yeir Weiss Research paper -https://www.cs.huji.ac.il/w~yweiss/iccv99.pdf

F.R.K. Chung. Spectral Graph Theory. American Mathematical Society, 19.

J. Shi and J. Malik. Normalized cuts and image segmentation.

Image Compression

https://www.researchgate.net/publication/321479958_Image_compression_using_singular_value_decomposition

https://in.mathworks.com/help/matlab/math/image-compression-with-low-rank-svd.html

http://www.ijmttjournal.org/Volume-65/Issue-8/IJMTT-V65I8P507.pdf

https://stackoverflow.com/questions/13614886/using-svd-to-compress-an-image-in-matlab

Deep-fakes.

https://arxiv.org/abs/1812.04948

https://www.warse.org/IJATCSE/static/pdf/file/ijatcse62922020.pdf

https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

https://towardsdatascience.com/vgg-neural-networks-the-next-step-after-alexnet-3f91fa9ffe2c#:~:text=VGG%20is%20an%20innovative%20object,most%20used%20image%2Drecognition%20architectures.