Computer Engineering Department

Linear Algebra

Project Report

Image Compression using SVD

Team Members:

Ahmad Waleed Akhtar	BSCE22003
Muhammad Arham	BSCE22007
Hadia Ahmad	BSCE22017

Introduction:

Our goal was to compress an image using SVD. Singular value decomposition (SVD) is a method of representing a matrix as a series of linear approximations that expose the underlying meaning-structure of the matrix. The goal of SVD is to find the optimal set of factors that best predict the outcome.

Application:

Following are some real life applications of SVD,

1, Low-rank approximation in image compression.

2, Low-rank approximation in recommender systems.

3, Finding the nearest orthogonal matrix to a given matrix in machine vision.

4, Principal component analysis (PCA).

5, Linear regression.

6, Signal processing.

7, Astrodynamics.

8, Numerical Weather Processing and etc.

Comparison:

Provided:

```
% function that takes in an image, N is how many singular values to use,
% and the show boolean should be defaulted to off but I use it in
% ImageTest.m to create a figure
function[error,finalImage] = rgbSVD(image,N,show)

clc

image=imread(image);

doubleR = zeros(size(image,1));
doubleG = zeros(size(image,1));
doubleB = zeros(size(image,1));

finalImage= repmat(0,[size(image,1) size(image,2) 3]);

for i = 1:size(image,1)
    for j = 1:size(image,2)
        doubleR(i,j) = image(i,j,1);
        doubleG(i,j) = image(i,j,2);
        doubleB(i,j) = image(i,j,3);
    end
end
```

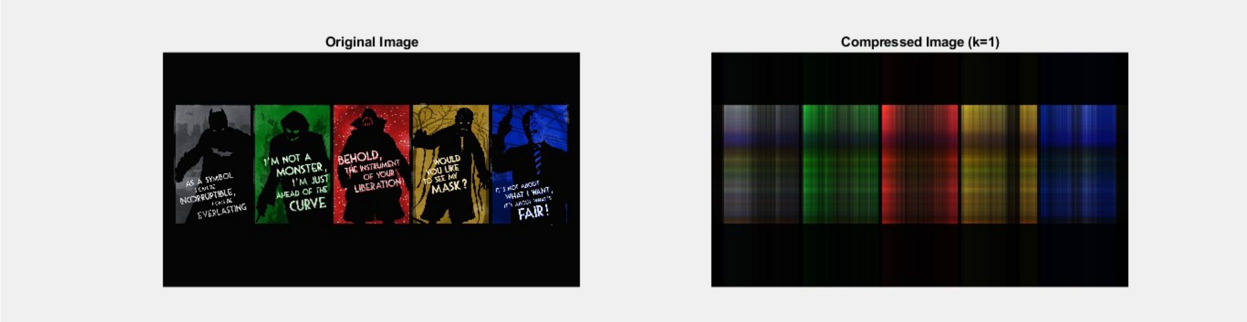
```
doubleR=im2double(doubleR);
doubleG=im2double(doubleG);
doubleB=im2double(doubleB);
%read the image and store it as matrix B, convert the image to a grayscale
%photo and convert the image to a class 'double'
[U,S,V]=svd(doubleR);
S(N+1:end,:)=0;
S(:,N+1:end)=0;
D=U*S*V';
finalImage(: , : , 1)=D;
 %Use singular value decomposition on the image doubleB, create a new matrix
 %C (for Compression diagonal) and zero out all entries above N, (which in
 %this case is 100). Then construct a new image, D, by using the new
 %diagonal matrix C.

[U,S,V]=svd(doubleG);
S(N+1:end,:)=0;
S(:,N+1:end)=0;
D=U*S*V';
finalImage(: , : , 2)=D;

[U,S,V]=svd(doubleB);
S(N+1:end,:)=0;
S(:,N+1:end)=0;
D=U*S*V';
finalImage(: , : , 3)=D;

if show ==1
    if N ~= 1
        figure(N-1);
    else
        figure(N);
    end

    imshow(uint8(finalImage));
    title(['Compressed Image with '  num2str(N)  ' singular values']);

end
%size(finalImage,(1))
%errordenom=size(finalImage,1)*size(finalImage,2)*3*256;
errordenom = 0;
errornum=0;
error = 0;
for i = 1:size(finalImage,1)
    for j = 1:size(finalImage,2)
        for k = 1:3

            errornum = double(errornum + abs((double(finalImage(i,j,k))-
double(image(i,j,k)))));
            errordenom = (256-double(image(i,j,k))) + errordenom;
            error = errornum/errordenom + error;
        end
    end
end
error = error * 100;
```

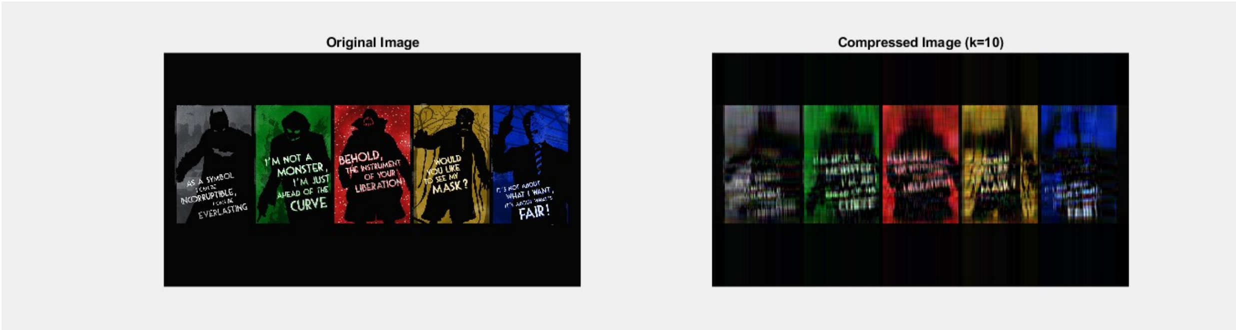
```
end
```

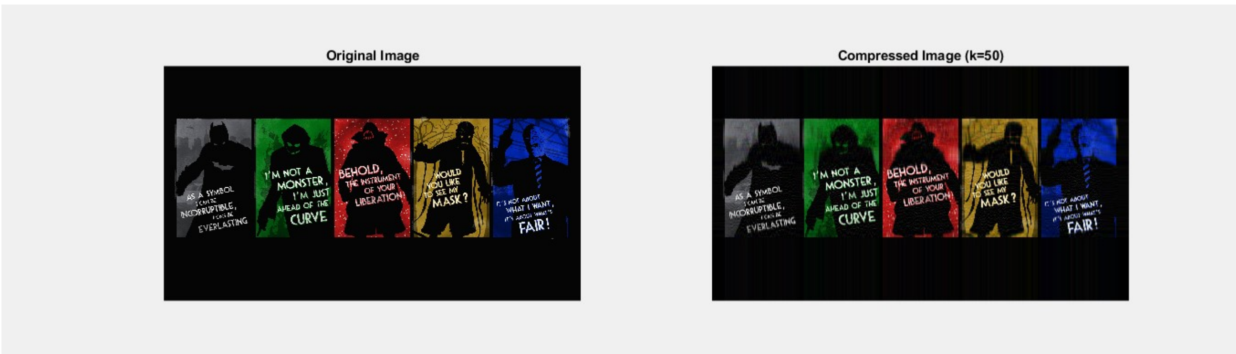
Our Code:

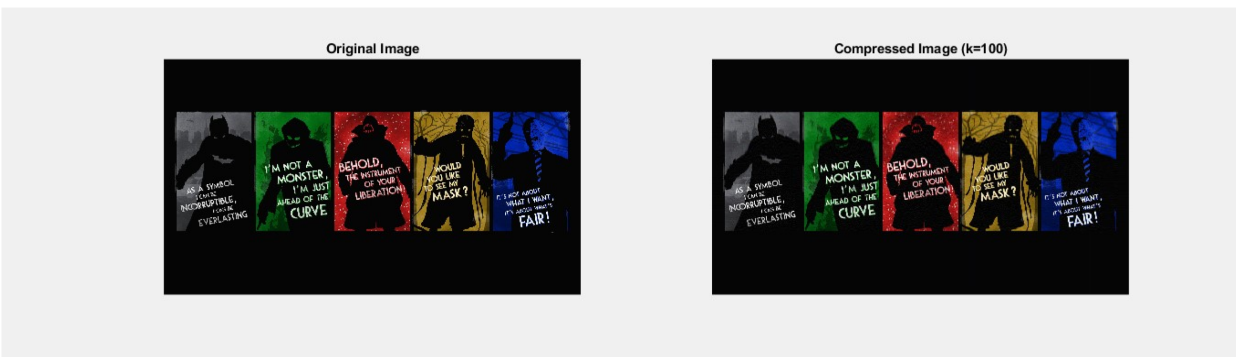
```
% Read the input image
img = imread('img.jpg');
sizeOrg = size(img);



% Split the image into red, green, and blue color channels
R = img(:,:,1);
G = img(:,:,2);
B = img(:,:,3);

% Compress each color channel using SVD
k = 100; % The number of singular values to keep
[UR, SR, VR] = svd(double(R));
[UG, SG, VG] = svd(double(G));
[UB, SB, VB] = svd(double(B));
compressedR = UR(:,1:k) * SR(1:k,1:k) * VR(:,1:k)';
compressedG = UG(:,1:k) * SG(1:k,1:k) * VG(:,1:k)';
compressedB = UB(:,1:k) * SB(1:k,1:k) * VB(:,1:k)';

% Combine the compressed color channels into a single compressed image
compressedImg = cat(3, uint8(compressedR), uint8(compressedG),
uint8(compressedB));

% Display the original and compressed images side-by-side
figure;
subplot(1,2,1);
imshow(img);
title('Original Image');
subplot(1,2,2);
imshow(compressedImg);
title(sprintf('Compressed Image (k=%d)', k));
```

Simulations:









Result Obtained & Conclusion:

By varying the value of K which represents the number of singular values being used to recreate image we were able to recreate a decent image of the original image. The image is compressed by successfully removing less important information and keep the more significant ones.