Image compression using singular value decomposition

IOP Conf. Series: Materials Science and Engineering **263** (2017) 042082 doi:10.1088/1757-899X/263/4/042082

Image compression using singular value decomposition

H R Swathi, Shah Sohini, Surbhi and G Gopichand
School of Computer Science and Engineering, VIT University, Vellore-632014, India

E-mail: gopichand.g@vit.ac.in

Abstract. We often need to transmit and store the images in many applications. Smaller the image, less is the cost associated with transmission and storage. So we often need to apply data compression techniques to reduce the storage space consumed by the image. One approach is to apply Singular Value Decomposition (SVD) on the image matrix. In this method, digital image is given to SVD. SVD refactors the given digital image into three matrices. Singular values are used to refactor the image and at the end of this process, image is represented with smaller set of values, hence reducing the storage space required by the image. Goal here is to achieve the image compression while preserving the important features which describe the original image. SVD can be adapted to any arbitrary, square, reversible and non-reversible matrix of m x n size. Compression ratio and Mean Square Error is used as performance metrics.

1. Introduction

As the technology is becoming an important part of day-to-day life, large volumes of data is being generated on a daily basis. This data can be of various forms, such as text, audio, video, images and so on. Images are one of the most popular data sharing method. Since 'A picture is worth a thousand words', we often exchange images to convey information. Revolution in the field of smart phones and smart devices has enabled the uses to use the pictorial data more conveniently. This way, we data scientists store, process, transmit huge amount of digital data on a daily basis. Storing the uncompressed data prove to be very expensive in terms of storage cost. Also, transmission of uncompressed data requires large bandwidth. Because of these reasons, techniques of compressing data before storage and transmission has become a significant area of interest in the research community; particularly in the fields such as artificial intelligence, pattern recognition, signal processing and so on. In this paper, we focus on reducing the storage space needed for storing an image file using the technique of singular value decomposition.

Image compression deals with reducing the size of data required to represent images. As the images are digital, goal of the image compression algorithm is to represent the image with lowest number of bits possible. Redundancies present in the image file can be exploited to in order to reduce the number of bits. Redundancy can be defined as approximate repetitive pattern that tend to give the degree of resolution to the image. However, quality of the image should not be compromised too much so that it becomes incomprehensible for the user. A good image compression algorithm should strike a balance between this tradeoff. Usually the requirement of the application dictate the tradeoff between compression and quality of the data.

Image compression techniques can be broadly classified into two categories. Lossless compression and lossy compression. Lossless compression techniques defines entropy, which limits the reduction in the image. It tends to produce the same copy as that of the original data. Lossless compression is known as reversible as it doesn't degrade the quality of the image. Lossy compression techniques identify the minute details and variations in the system which the human eye is not fine-tuned to recognize. Amount of storage required to store the image file can be reduced by eliminating such features. Lossy compression often compromises the quality of the image. However, it can be used to achieve the storage space requirements. Lossy compression is irreversible because it degrades the data. Compression possible in lossy techniques is much higher than lossless techniques in [11], [12] and [13].

Image is generally represented using a matrix, which is of size m×n. Here m is the number of rows, which signifies the pixel height of the image and n is the number of columns which signifies the pixel width of the image. Each element of the matrix gives the representation to each pixel that make up the image. In an image, darkness or brightness of a pixel relative to other pixels is represented by assigning a number to each pixel. In this way, each element of the matrix decides the characteristics of the corresponding pixel. In case of black and white images, each value in the matrix ranges between 0 (black) and 1 (one), which translates into the relative grayness of each pixel. Colored images are composed by three basic colors; red, green and blue (RGB). Every time a colored image is being stored, it is split by the computer into these three layers. Because of the additional layer involved, colored images consume more space when compared to grayscale images. So for a colored image, each pixel can be associated with three values, ranging from 0 (not colored) to 1 (saturated). These three values are assigned to red, green and blue. For example, if we are considering a 9 megapixel image, it can be represented by a 3000×3000 matrix. Considering a storage space of 1 byte per pixel, a grayscale 9 megapixel image needs 9MB storage space. Each pixel in the image matrix can be represented by an integer which falls between 0 and 255. If the image is colored, it will have three components (RGB). Each component needs to be represented by a separate matrix. Hence storing a colored image needs a storage space of 27Mb in this case.

In the sections below, compression scheme using SVD is explored in detail. Section 2 looks into the basic concepts of SVD in order to provide a perspective. Section 3 explores the literature that is available on the topic. Section 4 explains the proposed methodology. Section 5 narrates the results of implementation.

2. Background

One of the major factors that influence the compression is the presence of redundancy in the data. Redundancy can be of one of the following types:

- Coding Redundancy: Usage of less than optimal code words result in coding redundancy. Using lookup table for implementation makes the coding reversible. Examples of this technique are Huffman coding and Arithmetic coding.
- Inter-pixel Redundancy: Prediction of pixel values is based on values of its neighboring pixels. Original 2d pixel array is mapped into a different format. Other names of inter-pixel redundancy are spatial redundancy; inter frame redundancy, geometric redundancy.
- Psycho-visual Redundancy: Human eye is not fine-tuned to process every band of frequencies. Hence all the incoming data is not responded to with equal sensitivity. Some parts of the information will be more prominent than the others. This fact can be exploited when image redundancies are being removed. Psycho-visual redundancy makes use of this factor.

The SVD implementations try to take advantage of the redundancy as much as possible, with a goal of achieving smaller image. This can work well because awareness about redundant areas in the image helps the algorithm to eliminate only such area. This way, integrity of the image is not compromised.

2.1. Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) generally works by decomposing the original matrix. SVD aims to approximate the dataset of large number of dimensions using fewer dimensions. SVD considers a highly variable, high dimensional data points and exposes the substructure of the original data by reducing the higher dimensional data into lower dimensional data. Exposure of the substructure orders the data from most variation to the least. This helps to find the region of most variation and then later SVD can be used for reduction.

SVD is basically factorizing the given matrix into multiple matrices. SVD factorizes the given matrix with m number of rows and n number of columns it into three matrices; which can be written as $M = U\sum V^T$ where U and V^T are orthogonal matrices of the order m×m and n×n respectively and $\sum$ is a diagonal matrix of the order m×n. Diagonal matrix $\sum$ consists of nonnegative real numbers which are called as singular values of M. The m columns of U and n columns of V^T are known as left and right singular vectors of M respectively in [6] and [9].

SVD of a given matrix can be calculated as follows:

- From a given matrix M, calculate AA^T and A^TA.
- Use AA^T to form U, which is calculated by calculating eigen values and eigen vectors of AA^T.
- In the same way, V can be formed by calculating the eigen values and eigen vectors of A^TA.
- Columns of U and V are formed by dividing each eigen vector by its magnitude.
- Singular values are computed by taking the square root of eigen values. They are arranged in descending order in the diagonal matrix.

Figure 1 shows the matrices obtained after splitting the original image matrix

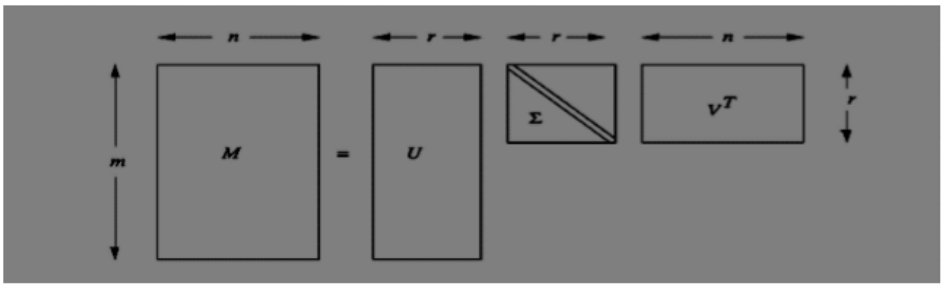

Figure 1. Finding the matrices $M = U \Sigma V^T$

2.2. Applying SVD in image compression

As we have seen above, SVD rewrites the original matrix as a sum of simpler matrices of rank one. When applied on a given image matrix, SVD decomposes it into three different matrices. However, application of SVD alone does not contribute for compression. After the SVD is applied from [7] and [8], most of the singular values will be discarded while few of them are retained. The fact that we arrange the singular values in the descending order of the diagonal matrix helps in this reduction. As the result of this arrangement, first value of the diagonal matrix contains most information and all the further values in the matrix contain decreasing amount of information about the image. So we can conclude that lover singular values contain negligible amount of information. Hence discarding such values reduces the size of the image while avoiding noticeable distortion from the original picture. Below steps formalized the procedure illustrated above.

Let A be the matrix of the given image. It can be expressed as

$A = U\sum V^T = \sum \mu_i u_i v_i^T$; summation ranging from 1 to r.

Expanding the summation,

$A = \mu_1 u_1 v_1^T + \mu_2 u_2 v_2^T + \ldots + \mu_r u_r v_r^T$

As we have seen, it is not necessary to compute the summation to the very last of the values in the matrices. Smaller values are dropped before the summation. After truncating the matrix, we get the following summation.

$A_k = \mu_1 u_1 v_1^T + \mu_2 u_2 v_2^T + \ldots + \mu_k u_k v_k^T$

Storage required by the reduces matrix A_k is k(m+n+1) units. Value of k will be closer to n, but less than n. Hence resulting image will have a good resolution and it is not distorted too much. It is evident that storage space required for the compressed image depends on the value of k. So the value of k can be adjusted as per the requirement. This process is illustrated in Figure 2.

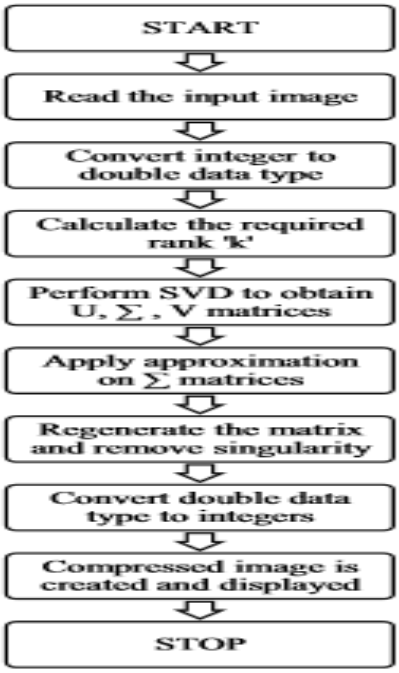

Figure 2. Image compression using SVD

3. Literature Survey

S K Singh et al. [1] has implemented compression of image. Image matrix is processed using the technique of Singular Value Decomposition (SVD). This technique carries out the compaction in according to the compaction of energy. Moreover, it concentrates on initial few columns which tend to have the localized content of the matrices. Overall energy of the image is represented by singular values of the image. Storage and transmission of image demands compression to be carried out on the original image. To resolve this problem, image is seen as matrix and compression of matrix is performed by performing various operations on matrix. The SVD can be implemented on matrices of varying sizes and types such as arbitrary, square, reversible and non- reversible matrices. SVD reduces the storage space of an image. Effect of rank is also shown in SVD decomposition that will judge the quality of compressed image.

M. E. I. Tian et al. [2] has proposed three techniques for compressing the images. They have also studied and analyzed the efficiency of SVD based image compression techniques. This paper has introduced a technique wherein efficient representation of singular vectors in each sub-block is used. This approach is called as adaptive singular value selection scheme. Three schemes are-Direct compression and decomposition scheme, adaptive singular value selection scheme, singular value subtracting one update scheme. In Direct scheme, original matrix is approximated using singular values which are fewer. As opposed to compressing the entire image in one go, this selection scheme tries to divide the original image into sub-block which are smaller in size with the goal of working with these smaller images. This approach is chosen with the aim of using the uneven complexity present in the original image to our advantage.

P. Wadem et al. [3] has proposed the changing the hybrid KLT-SVD system to KTL coding system. KLT stands for Karhunen-Lobve transform. An image contains variations in the statistics which are local to the image. We can exploit this fact and use the transformation adaptive technique to

transform the coding of images. This technique uses the similarities between SVD and KLT and due to this coding efficiency of the technique is also improved. If the purpose is to use the transform on all the blocks which builds the image, KLT can be used efficiently. However, improves compression for a given block is achieved by SVD. So in order to use this advantage, a system is built that forms the SVD transforms and creates the approximations using them. Lagrange technique can be used to reduce the distortion of the image during reconstruction. This can be used for switching to KLT. But the limitation here is number of bits used for reconstructing every block is very high. This increases the cost associated with this technique. This can be improvised by using substitute methods.

T. J. Peters et al. [4] has implemented SVD (singular value decomposition) to compress the microarray image. Huge amounts of DNA information for research purposes are stores as microarray images. These are of high resolution images which highlights minute details of the image. Because of the high resolution, these images tend to be larger in size, which means storage on the hard disk requires lot of space. So it is very important to reduce the size of the image without compromising the quality or compromising the amount of detail present in the image. This calls for comparatively complicated process, where in microarray images need to be clustered and classified before selecting the features. SVD can be used here to divide the image into small sub -images and on each sub-image SVD is performed. This method gives a better high peak signal to noise ratio in addition to increasing compression ratio.

H. S. Prasantha et al. [5] have implemented another approach for compression of images using singular value decomposition (SVD). By applying singular value decomposition on the image matrix, compression of image is achieved .SVD takes into consideration. This method employs a data compression technique, wherein operations are performed on matrix after the image is decomposed into matrices. For every image, matrix is found and as per points graph is plotted for dissimilar ranks in [10]. Further, it is observed that as rank increases in image matrix, the number of entries will also increases which leads to make picture quality better correspondingly. Due to this, more compression ratio has been achieved by smaller ranks.

4. Proposed solution

Image matrix composed of RGB signals. For the purpose of storing and transmitting the image, RGB representation is not optimal. This is because of the presence of redundant information. So the proposed solution looks to improve this by transforming the RGB spectrum into YC_BC_R spectrum. Y represents the brightness. Alternatively, luminance can also be used. C_B and C_R represent blue and red chrome components. This new representation provides an approximation to the processing, where primary colors are processed into perceptually meaningful information. This representation segregates the Y components into two Chroma components. These components can be stored with higher resolution or transmitted at higher bandwidth. They can also be treated separately, by compressing or reducing the bandwidth.

In the proposed system involves using YC_BC_R format for representing original image which is in RGB representation. Then constituent components can be obtained by breaking down the YC_BC_R representation. Hence we get 3 components corresponding to Y, C_B, and C_R. SVD is then applied on these components. Their respective U, $\sum$ and V components are obtained; namely, U_y, $\sum_y$, V_y corresponding to y, U_b, $\sum_b$, V_b corresponding to C_B and U_r, $\sum_r$, V_r corresponding to C_R. Since U and V represent the spatial attributes of the image, they can be taken from any color space. However, since Y represents the brightness, it has to be used for all three components. Loss of information can be achieved using this approach. Since Y is being used in all three components, we have chosen to use U and V of Y component. All three components of the image is used in reconstruction so as to reduce the damage due to loss. Frequency components for U and V are generated and thresholding technique is used to remove the low value SV from the matrix. This method helps to achieve great deal of compression while retaining the quality. Image is reconstructed using $\sum$,U and V of all the components.

4.1. Algorithm:
INPUT: Image matrix I
OUTPUT: Compressed image I'
Steps:

1. Obtain YC_BC_R from RGB.
2. Obtain the constituent components Y, C_B, and C_R.
3. Generate three vectors for each of the components by applying SVD.
4. Obtain the intermediate image by applying the reconstruction algorithm to the components.
5. Obtain the frequency components for U and V components.
6. Obtain the resultant image using the threshold values of U and V matrices.
7. Final image is formed by concatenating the components Y, C_B, and C_R.
8. Calculate the compression ratio.

This algorithm is illustrated in Figure 3. We wish to implement this algorithm on R programming platform.

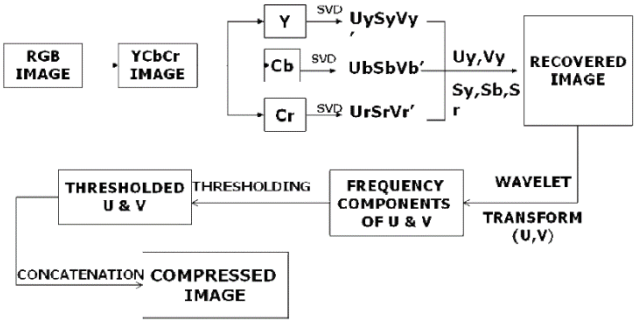

Figure 3. Proposed algorithm

5. Results

This algorithm has been implemented in R programming platform. An image of size 700Kb and is considered. It is run through the algorithm, considering different number of columns (k) each time. Size and the resolution of output image is evaluated each time.

Table 1. Compressed image sizes corresponding to different values of k.

K	CR	Size of compressed image(in Kb)
5	174.55	106.6
10	87.27	124.3
40	21.81	159
80	10.9	176.1
100	8.72	180
800	1.09	215.8

5.1. Performance metrics
Here we consider two metrics as evaluation criteria.

- Compression ratio: It is the ratio of memory required to store the real image to the memory required to store new image (changed).it calculates the extent to which pixels in image are changed (compressed). It is calculated as $CR = m \times n/(k \times (m+n+1))$.

- Mean Square Error (MSE): Tells the extent to which compressed image quality is deteriorated when matched with real image quality. It is the square of the distance between bit values of the real image to the bit value of the new image (compressed).

Comparison of different runs is depicted in table 1. Figure 4 shows the compression of images with different values of k.

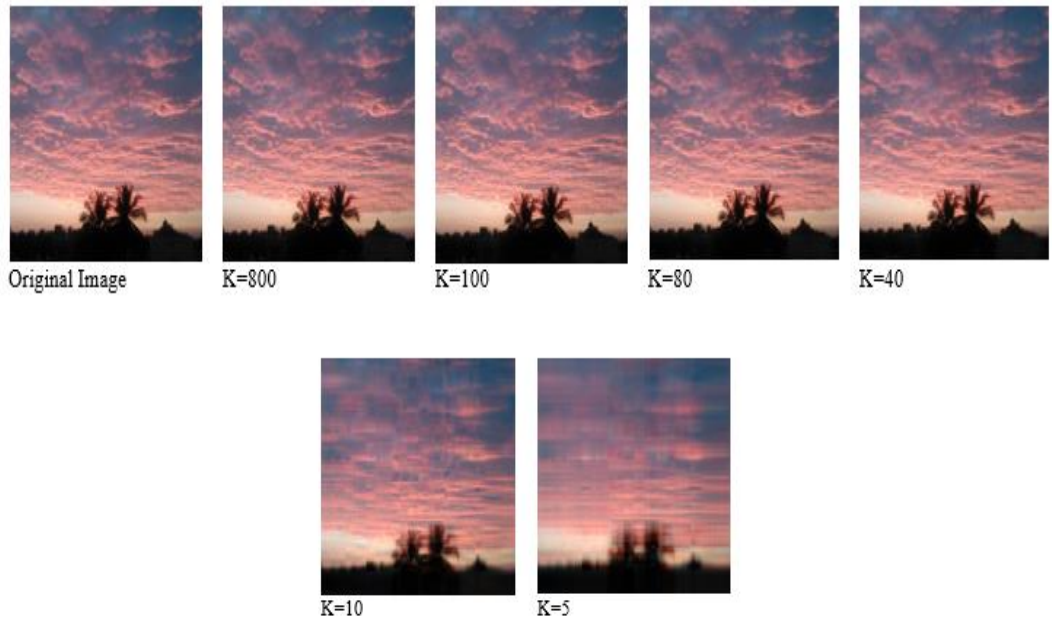

Figure 4. Compressions for different values of k.

6. Conclusion

From the above results, we can see that compression required by the application can be achieved by choosing an appropriate value of k. It is observed that with the increasing rank of the matrices, maximum values are concentrated in initial coefficients. Number of entries in the matrix increases with the increasing rank in the reconstructed matrix. Another advantage of using SVD is the less computational complexity involved in achieving the compression. However, scaling could be a challenge in this approach as the process of compression is sequential. This can be improved by parallelizing the computation of SVD for R, G and B matrices.

References

[1] Singh S K and Kumar S 2009 A Framework to Design Novel SVD Based Color Image Compression 1–6
[2] Tian M E I, Luo S, and Liao L 2005 An investigation into using singular value decomposition as a method of image compression 18–21
[3] Wadem P and Ramstad T A 1997 Hybrid klt-svd image compression 2713–2716
[4] Internationale C, Peters T J, Smolíková-wachowiak R and Wachowiak M P 2007 Microarray Image Compression Using a Variation of Singular Value Decomposition 1176–1179
[5] Prasantha H S and B M K N 2007 Image compression using SVD 143–145
[6] Kotera H RGB to spectral lmage conversion using spectral pallete and compression by svd 461–464
[7] Dixit P M M and Kulkarni P K 2012 Variable Scaling Factor based Invisible Image Watermarking using Hybrid DWT - SVD Compression - Decompression Technique 2–5.

[8] Rufai A M, Anbarjafari G and Demirel H Huffman Coding and Singular Value Decomposition
[9] Jabbar S 2015 Using Approximate K-SVD Algorithm
[10] Ayu P, Savitri I, Murdiansyah D T and Astuti W 2016 Digital Medical Image Compression Algorithm Using Adaptive Huffman Coding and Graph Based Quantization Based on IWT-SVD **4**
[11] kumar S V and Santhi V 2016 Neuro-Fuzzy Approach for Speckle Noise Reduction in SAR Images 251-260
[12] Ali S S 2015 A Novel Image Quality Assessment Metric using Singular Value Decomposition 170–173
[13] Aishwarya K M, Ramesh R, Sobarad P M and Singh V 2016 Lossy Image Compression using SVD Coding Algorithm 1384–1389