Image Compression using SVD

https://www.notion.so/Image-Compression-using-SVD-c0135aab5d9f47a080dca063f7f456c0#6477a847011a48319cddb05d8353ba86

Supervisor:	**Posted:**	**Course:**
Dr. Tsui-Wei Weng	Nov 3, 2022	DSC 210 FA'22 Numerical Linear Algebra

Team:

Yashi Shukla	Sagarika Sardesai	Shreya Reddy Pakala	Sai Kaushik Soma
PID: A59020018	PID: A59020049	PID: A59019863	PID: A59020013
Data Science	Data Science	Data Science	Data Science

1. Introduction

Image compression is the process of minimizing the image size in bytes without degrading image quality and is of 2 types - Lossy and Lossless.

In Lossy compression, non critical and redundant data is permanently removed. Lossy is generally used where information loss is acceptable and algorithms like DCT and SVD fall under this. The drawbacks of this method include image degradation after too much of compression and the inability to reconstruct the original image from the compressed one.

In Lossless compression, images are compressed without removing critical data or reducing image quality. The original image can be reconstructed from the compressed image. Lossless is applied where image quality is crucial and algorithms like Huffman Coding , Arithmetic encoding fall under this. A major drawback of this is that the compressed image size is still quite large.

1.1 Applications:

Image compression is used extensively in :

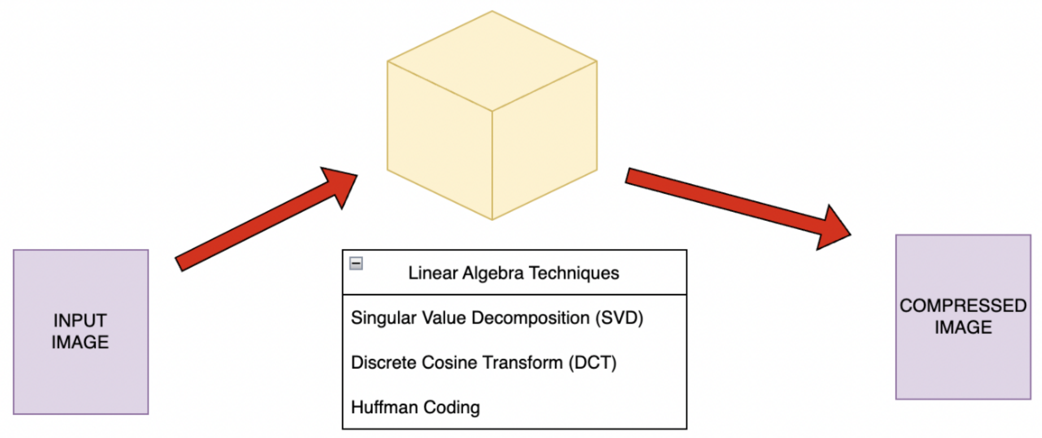
- Pattern recognition
- Computer/Machine Vision
- Video Processing

Some of the real world applications that utilize image compression involve:

- Websites
- Digital Photography
- Image editing softwares
- Health industry
- Defense and Security
- and many more.

2. Problem Formulation

2.1 Relation to numerical linear algebra:


INPUT IMAGE
Linear Algebra Techniques
Singular Value Decomposition (SVD)
Discrete Cosine Transform (DCT)
Huffman Coding
COMPRESSED IMAGE

Every image consists of a matrix of pixels which is passed through one of the Linear Algebra Techniques and then we obtain a compressed image. Techniques like Singular Value Decomposition (SVD), Discrete Cosine

Transform (DCT), and Huffman Coding are all built using Numerical Linear Algebra whose functionalities are utilized for image compression.

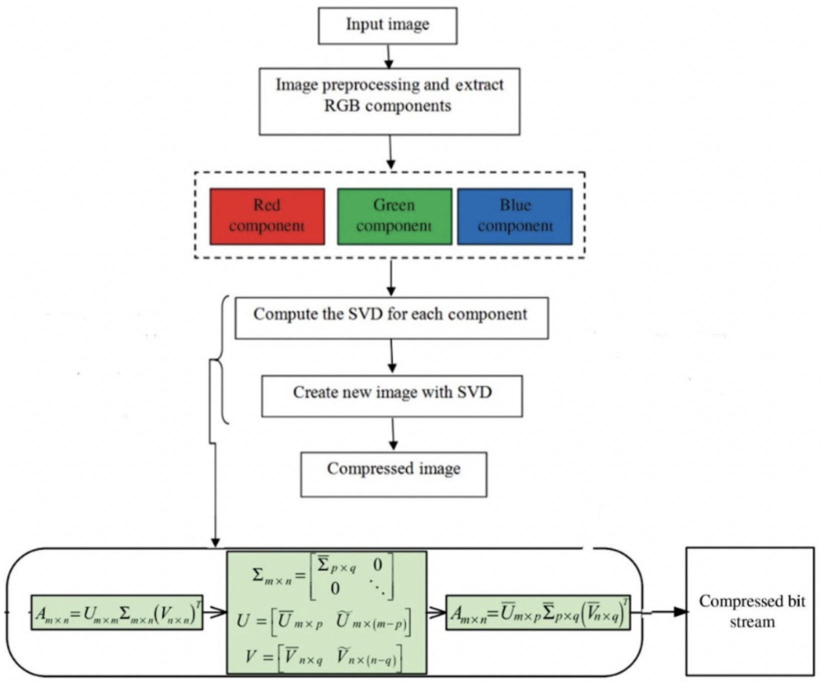
2.2 Approach description:

Singular value decomposition is a factorization of a real or complex matrix. SVD aims to approximate a dataset of large number of dimensions using fewer dimensions. SVD considers 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 the most variation to the least. It generalizes the eigen decomposition of a square normal matrix with an orthonormal eigen basis to any $m \times n$ matrix. It is represented by the equation:

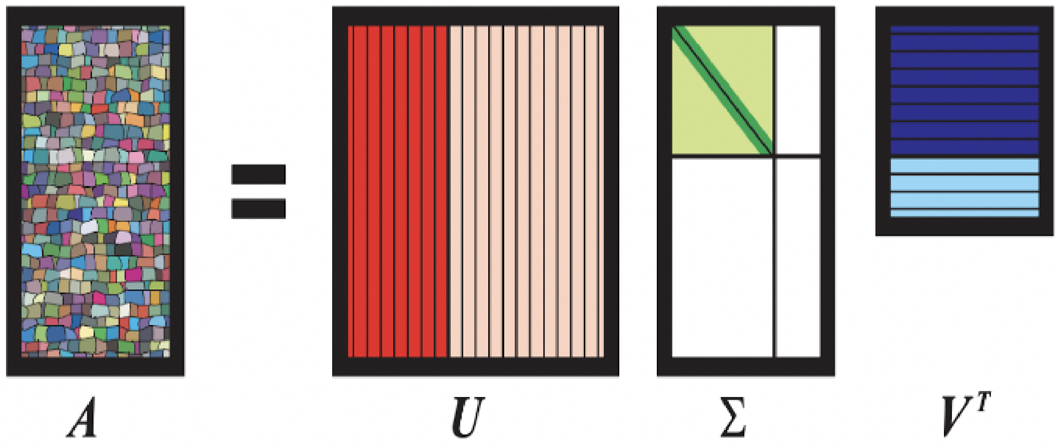
$$\underset{(m\times n)}{A} = \underset{(m\times m)}{U} \underset{(m\times n)}{\sum} \underset{(n\times n)}{V}^{T}$$

- Let A be an $m \times n$ matrix with rank r. Then there exists an $m \times n$ matrix $\sum$ for which the diagonal entries are the first r singular values of A where $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0$, an $m \times m$ orthogonal matrix U and an $n \times n$ orthogonal matrix V.
- Calculation of the SVD consists of finding the eigenvalues and eigenvectors of AA^T and A^TA.
The eigenvectors of A^TA make up the columns of V, the eigenvectors of AA^T make up the columns of U. Also, the singular values in $\sum$ are square roots of eigenvalues from AA^T or A^TA.
- The singular values are the diagonal entries of the $\sum$ matrix and are arranged in descending order ($\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r$). The singular values are always real numbers. If matrix A is a real matrix, then U and V are also real.

Considering the application of SVD in image compression using the following flowchart:


Input image
Image preprocessing and extract RGB components
Red component
Green component
Blue component
Compute the SVD for each component
Create new image with SVD
Compressed image
$A_{m\times n}=U_{m\times m}\Sigma_{m\times n}(V_{n\times n})^T$
$\Sigma_{m\times n}=\begin{bmatrix}\overline{\Sigma}_{p\times q} & 0\\ 0 & \ddots\end{bmatrix}$
$U=\begin{bmatrix}\overline{U}_{m\times p} & \widetilde{U}_{m\times(m-p)}\end{bmatrix}$
$V=\begin{bmatrix}\overline{V}_{n\times q} & \widetilde{V}_{n\times(n-q)}\end{bmatrix}$
$A_{m\times n}=\overline{U}_{m\times p}\overline{\Sigma}_{p\times q}(\overline{V}_{n\times q})^T$
Compressed bit stream

First, the input image undergoes image processing from which we extract RGB (Red, Blue, Geen) components. On each of the components, SVD is computed where the image undergoes SVD matrix factorization. This can be seen in the figure given below where SVD refactors the given image into three matrices. Singular values are used to refactor the image and at the end of this process, the image is represented with a smaller set of values, hence reducing the storage space required by the image. The color gradient in each of the matrices as seen in the image signifies the singular values.



Once the SVD is computed the final output is a compressed bit stream which is later converted to a compressed image.

---

3. State-of-the-art

The state-of-the-art technique that we will be implementing and comparing the SVD approach with, is Discrete Cosine Transform (DCT). Like SVD, DCT is also a lossy compression and is generally used to compress JPEG Images.

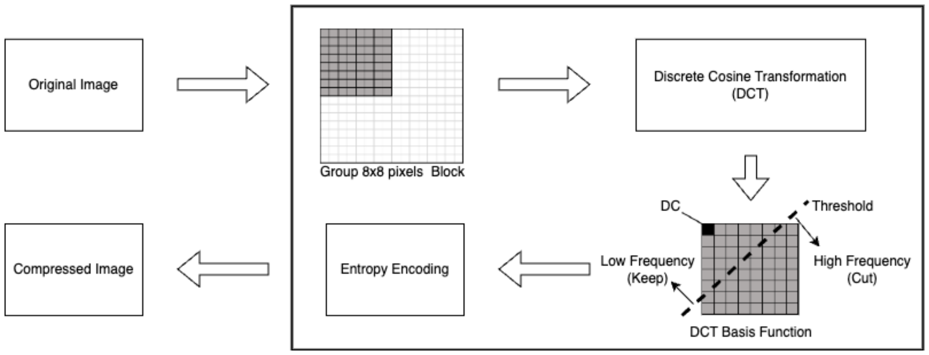
3.1 SOTA Approach description:

DCT in itself is a fast computing Fourier Transform which maps the signals to corresponding values in the frequency domain. It transforms them to a domain in which they can be more efficiently encoded.

This algorithm is widely used in image compression. For multichannel images, we apply DCT individually to every channel.

Steps for implementing DCT for image compression:

- DCT breaks the image into n x n blocks, usually the value of n=8 according to JPEG standards.
- DCT is then applied to each of the block serially; the values which fall under the frequency domain are kept and other values are cut down.
- Lastly, each block goes through entropy encoding and we get our compressed image.

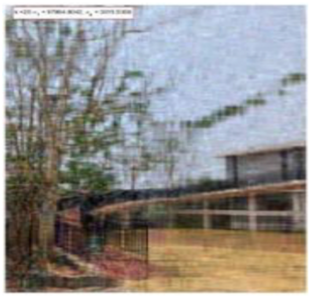


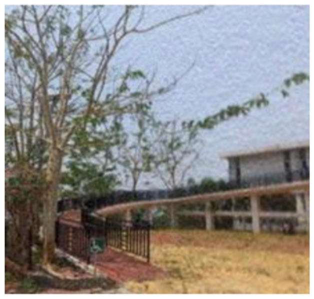
3.2 Why SOTA?

For compression rates up to 30%, both SVD and DCT yield no visual difference in the compressed image. But DCT is more acceptable and widely used technique in image compression compared to SVD for the following reasons:

- DCT spreads its loss throughout the image evenly unlike SVD where we can actually lose an entire vector of data at a time.
- As DCT is a block based approach, it takes less computing memory while processing the images. But SVD works on the whole image at once and needs more memory.
- For higher compression ratios, DCT provides a less distorted image as compared to SVD.

Original Image (174 KB)


SVD (79 KB)


DCT (75 KB)

4. Experimental setup:

4.1 Dataset:

The Images 4k dataset consists of 2057 jpg images.

4.2 Libraries and Tools:

We will implement our project in **Python**, using the following libraries:

- **Matplotlib:** Matplotlib is a multi-platform, graphical plotting library for Python used to create interactive visualizations.
- **Numpy:** Numpy is a Python library used for working with arrays. It provides support for large, multi-dimensional arrays and matrices as well as computations including statistical functions, linear algebra, Fourier transform, and other mathematical functions to utilize on these arrays.
- **OpenCV:** OpenCV is an open-source computer vision library mainly used for image processing and machine learning applications. It provides functions to read, save and manipulate image properties among several others.

4.3 Project Logistics:

- **GitHub:** Open source platform used for version control and collaboration.
- **Notion:** To create the project proposal document.
- **Google Colab:** Jupyter notebook environment for collaborating on notebooks.

4.4 Performance Metrics:

The performance metrics that we'll be using are described below:

- **Compression Ratio:** Compression Ratio is the ratio between the sizes of the original image and the compressed image. Higher the compression ratio, smaller the size but lower the quality of the compressed image.

$$Compression\ Ratio\ =\ \frac{Size\ of\ the\ Original\ Image}{Size\ of\ the\ Compressed\ Image}$$

- **Mean Squared Error (MSE):** Mean Squared Error is the average of squared errors where error is the difference between each corresponding pixel in the actual and compressed image. Lower the MSE, closer the compressed image is to the original image.

$$MSE\ between\ two\ images\ I(x,y)\ and\ I'(x,y):$$
$$MSE\ =\ \frac{1}{M \times N}\sum_{y=1}^{M}\sum_{x=1}^{N}[I(x,y) - I'(x,y)]^2$$

- **Peak Signal to Noise Ratio (PSNR):** PSNR measures the ratio between maximum pixel intensity and the power of distorting noise affecting the quality of its representation. The noise is the error produced by compression. Higher the PSNR value, better the quality of compressed image.

$$PSNR = \frac{10\log_{10}(peakval)^2}{MSE}$$
$$where\ peakval = maximum\ intensity\ value\ of\ the\ image$$
$$and\ MSE = Mean\ squared\ error$$

- **Structural Similarity Index Measure (SSIM):** In this method, the structural similarity index calculation is based on three aspects namely: image luminance, contrast and structure. The index measures similarity between two images x and y, which can be calculated using the given formula:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_x\sigma_y + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$
$$where\ \mu_x, \mu_y\ are\ the\ local\ means;$$
$$\sigma_x, \sigma_y\ are\ the\ standard\ deviations;$$
$$C_1, C_2\ are\ the\ constants\ used\ to\ ensure\ stability\ when\ denominator\ is\ close\ to\ 0$$

5. Conclusion

Our next steps in the project include exploring the Images4k dataset, applying the SVD technique on this dataset to obtain compressed images, and applying the State-of-the-art technique, DCT, on the same data set. We will also compare the compression results of both SVD and DCT using the performance metrics described in section 4.4.

We expect to see reduced image sizes post-compression. Also, we would like to show that increased compression results in a more distorted image. Furthermore, we expect to see DCT resulting in less distorted images as compared to SVD.

6. References

- Sara, U. et al. (2019) *Journal of Computer and Communications*, **7**, 8-18 Image Quality Assessment through FSIM, SSIM, MSE and PSNR—A Comparative Study.
- Asnaoui, K. (2020)*Journal of Intelligent Systems*, 29(1), 1345-1359 Image Compression Based on Block SVD Power Method.
- H R Swathi *et al* 2017 *IOP Conf. Ser.: Mater. Sci. Eng* **263** 042082 Image compression using singular value decomposition
- Medium. 2020. *Compressing Images Using Linear Algebra.* [online] Available at: https://medium.com/analytics-vidhya/compressing-images-using-linear-algebra-bdac64c5e7ef
- Rebaza J (2012). *A First Course in Applied Mathematics.* Image Compression. Wiley & Sons Ltd. http://people.missouristate.edu/jrebaza/assets/10compression.pdf
- Singular Value Decomposition - Applications in Image Processing. [online] Available at: https://www2.karlin.mff.cuni.cz/~tuma/Aplikace15/prezentace_Hnetynkova.pdf
- Singular Value Decomposition Tutorial. [online] Available at: https://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm