

Introduction

Compression of images is an incredibly important process which occurs millions of times every day, images by themselves are simply too large to be shared or stored efficiently despite the advancements in storage due to the fact that they are essentially a 2D matrix of bits it's still too large.

That is the reason people are continuously developing better algorithms for compression of the image as well as maintaining a low loss percentage. This project's aims are to explore efficient algorithms that can be used for this same process.

Review of Literature

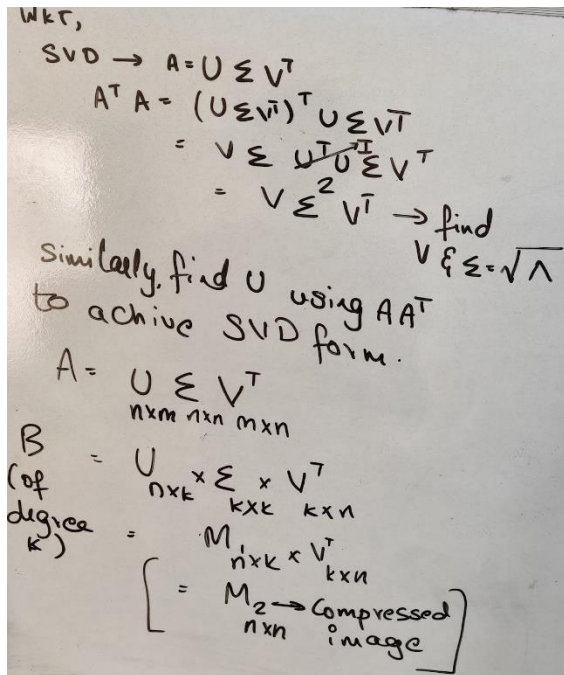
This topic has been done several times by various academics that have far greater experience in the field, to the extent that this algorithm has been replaced by far better ones, that have been developed by these professionals.

Even so, this algorithm has also been developed to be more complicated in a way to improve its performance. This project made us learn more about compression, as well as the use of SVD; utmost this is a good example of its use in image scaling.

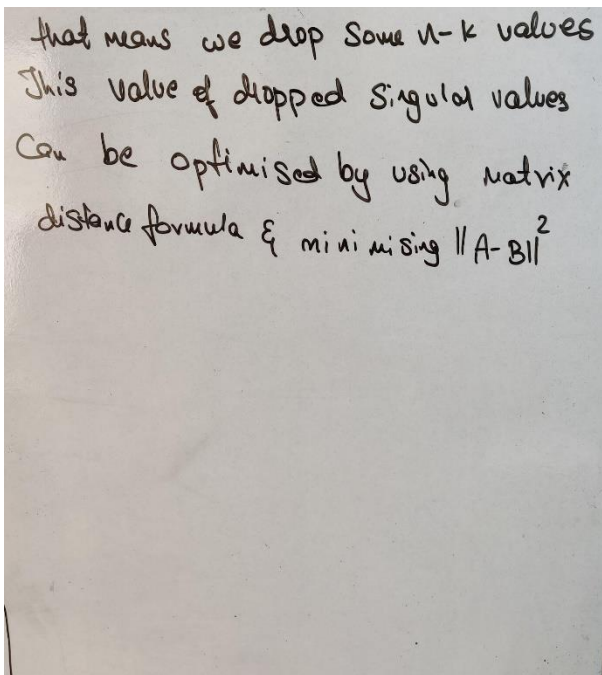
In this project we created a no-frills and very simple SVD image compression script with intention of converting it into a service

Report on the Present Investigation

In the scope of this project, we applied SVD to images converted to matrices and later dropped trailing singular values to cause “lossy” compression. A few write-ups in this regard:



WKT,
 $SVD \rightarrow A = U \Sigma V^T$
 $A^T A = (U \Sigma V^T)^T U \Sigma V^T$
 $= V \Sigma^T U^T U \Sigma V^T$
 $= V \Sigma^2 V^T \rightarrow \text{find } V \text{ \& } \Sigma = \sqrt{\Lambda}$
Similarly, find U using AA^T
to achieve SVD form.
 $A = U \Sigma V^T$
 $n \times m \quad n \times n \quad m \times n$
 $B = U \Sigma V^T$
 $n \times k \quad k \times k \quad k \times n$
(of degree k)
 $= M_1 \times V^T$
 $n \times k \quad k \times n$
 $\left[= M_2 \rightarrow \text{Compressed image} \right]$
 $n \times n$



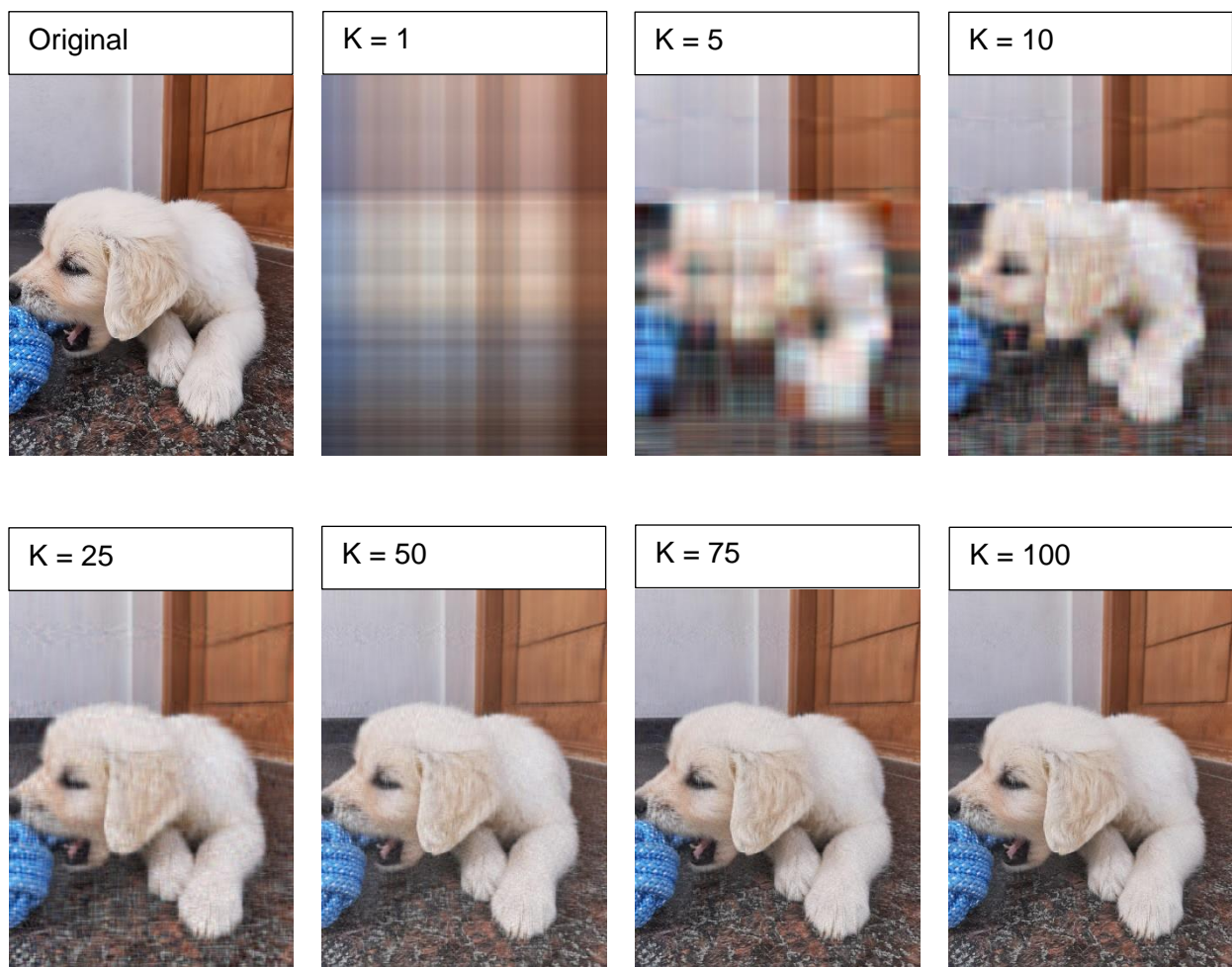
that means we drop some $n-k$ values
This value of dropped singular values
can be optimised by using matrix
distance formula & minimising $\|A-B\|^2$

Results and Discussions

An initial image of size 142217 Bytes of dimensions 960 x 1280 was passed through the compression algorithm for various values of different number of columns (k). The size of each of the obtained output images are recorded.

```
navin@usermachine:~/github/Image-Impression(main) » python3 compressor.py -v 125 images/test.jpg comp_test_500.jpg
Compressing...
k : 125
(1280, 960, 3)
Initial number of singular values : 960 960 960
125
125
125
(125,)
Lossy conversion from float64 to uint8. Range [-0.05325188521596042, 1.044267037303253]. Convert image to uint8 prior
ppress this warning.
Initial image size : 142217
Output image size : 135405
Compression ratio : 1.0503083342564898
Image compressed successfully!
```

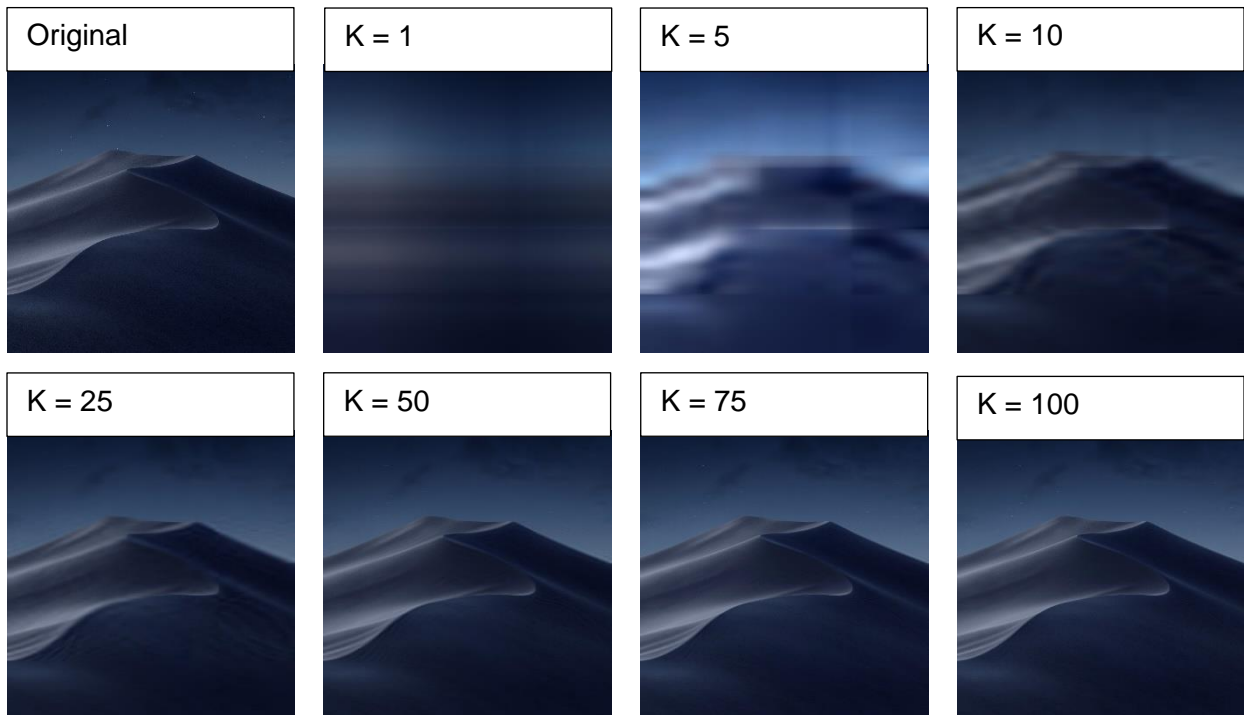
K	COMPRESSION FACTOR	SIZE OF COMPRESSED IMAGE (Bytes)
1	3.12	45578
5	2.17	65525
10	1.84	77284
25	1.44	98740
50	1.26	112983
75	1.15	123735
100	1.09	129201



Square Images

An initial image of size 883866 Bytes of dimensions 2048 x 2048 was passed through the compression algorithm for various values of different number of columns (k). The size of each of the obtained output images are recorded.

K	COMPRESSION FACTOR	SIZE OF COMPRESSED IMAGE (Bytes)
1	11.19	78990
5	8.43	104908
10	9.53	92684
25	8.49	104019
50	7.44	118760
75	6.08	145386
100	5.29	167051



Summary and Conclusions

We observed very small compression ratios, nothing too crazy in the results, this may be due to the fact that we are already using compressed JPG and PNG formats. Anyhow, even using simple SVD compression we can make our compression ratios can be made better by using a matrix distance finding algorithm which is also a future scope (along with services) for this mini-project.

Bibliography

1. <https://ieeexplore.ieee.org/abstract/document/1093309>
2. Image Compression using Singular Value Decomposition
3. Lossy image compression using singular value decomposition and wavelet difference reduction - ScienceDirect
4. <https://www.cmi.ac.in/~ksutar/NLA2013/imagecompression.pdf>
5. <https://web.stanford.edu/class/cme335/lecture6.pdf>
6. https://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm