

Singular Value Decomposition for Image Compression

James DeGruccio

July 27, 2024

Introduction

Singular Value Decomposition (SVD) is the factorization of any $m \times n$ matrix, A , into $U\Sigma V^T$. In this factorization, Σ is a diagonal $m \times n$ matrix composed of the eigenvalues shared between AA^T and A^TA . U and V are $m \times m$ and $n \times n$ orthogonal matrices used with Σ to produce $A = U\Sigma V^T$.

The matrix, A , can also be factored into a Compact/Reduced SVD^[1]. The Compact/Reduced SVD shrinks size of the Σ matrix by truncating the zero rows and columns in the Σ matrix. The corresponding columns of U and rows of V are also removed, reducing the amount of data needed to store the Decomposition. Formally, the Compact/Reduced SVD is denoted as $A = U_r\Sigma_rV_r^T$, where r is the rank of matrix A , U_r is $m \times r$, V_r is $n \times r$, and Σ_r is $r \times r$. If $r \leq m, n$, the Compact SVD could losslessly store a matrix with less space compared to the raw matrix itself.

A Truncated SVD^[1] can compress a matrix further by removing information from the matrix. This compression involves removing non-zero entries in Σ and the corresponding columns of U and rows of V .

Since an image can be represented by a matrix of pixel values, an image can be factorized using Singular Value Decomposition. The factored SVD matrix can then be truncated to implement a lossy compression algorithm.

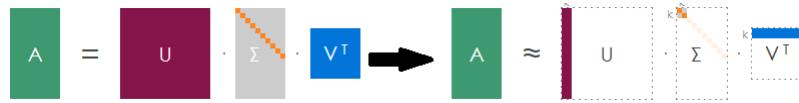


Figure 1: Truncated SVD Illustration^[2]

Method

A Python program was written to implement the SVD compression algorithm. The program takes two command line arguments:

1. Input file name
2. Size of truncated Σ array.

To perform image compression, the Pillow Python library is used to read image files and NumPy is used to perform the Singular Value Decomposition. The A , U , Σ , V matrices are organized in a class named *SVD_factored*. This class has two methods, *SVD_Compact* and *SVD_Truncate*, which calculates the Compact SVD and Truncated SVD respectively. NumPy generates U , Σ , and V matrices that are sorted from greatest to least with respect to Σ 's singular values. This makes the Compact SVD and Truncate SVD algorithms incredibly simple, since all there is to do is truncate the last rows/columns of the matrices. For Compact SVD, keep the first r vectors. For Truncate SVD, keep the first n vectors, where n is specified in command line arguments.

Results

Since I did give myself time to create a image file format for the SVD compression, there isn't a way to directly compare the actual compression ratios resulting from this algorithm. Tim Baumann has a general formula on his website to calculate the compression ratio [2]. If w is the image width, h is the image height, and n is truncated size of the singular values, the compression ratio is given below:

$$\frac{w \times h}{w \times n + n \times h}$$

I performed the SVD compression algorithm on a picture of my parent's dog. The results of this for various values of n can be seen below the references section.

References

- [1] Brigham Young University. (n.d.). The SVD and image compression. <https://acme.byu.edu/00000179-aa18-d402-af7f-abf806b60000/svd2020-pdf>
- [2] Baumann, T. (n.d.). Image compression with singular value decomposition. SVD-Demo: Image Compression. <https://timbaumann.info/svd-image-compression-demo/>



Figure 2: Original picture of Loki.

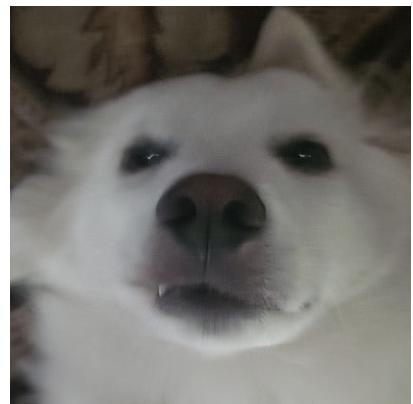


Figure 3: Compressed with $n=50$

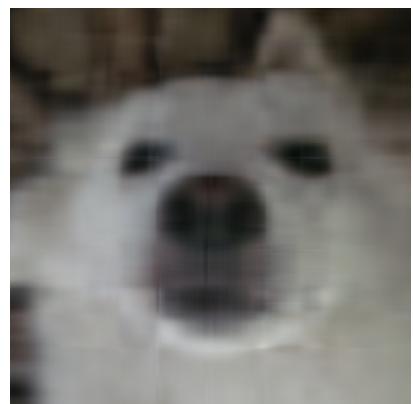


Figure 4: Compressed with $n=10$



Figure 5: Compressed with n=5

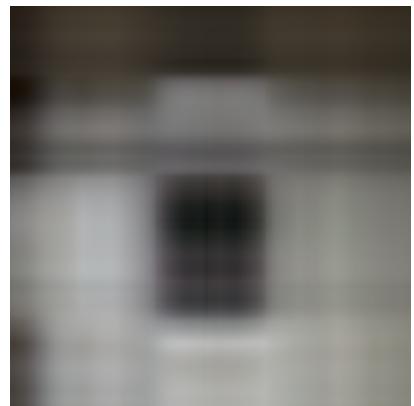


Figure 6: Compressed with n=2

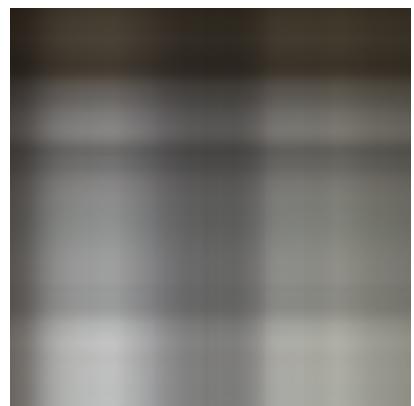


Figure 7: Compressed with n=1