Image Compression Using Singular Value Decomposition

Antonio Perez and Raymundo Camacho

ABSTRACT

The Singular Value Decomposition [SVD] is a nifty tool, often present behind processes like formulating Netflix show recommendations, constructing facial recognition algorithms, and working with the compression of images. SVD allows for a reduction in an image's byte size by decomposing a given image and rendering another that is nearly indistinguishable from the original, but with the added benefit of taking up less storage space. In this report, the underlying linear algebra concepts behind SVD, and its applications to image compression, will be discussed and examples of implementation will be provided.

I. INTRODUCTION

The underlying geometric interpretation behind SVD takes a matrix and has it map a unit sphere onto an ellipsoid, both in $\mathbb{R}^n$.

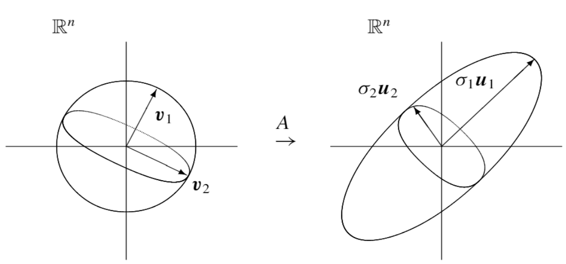

Fig. 1: Map that sphere!

Where $\sigma_1 \geq \sigma_2 ... \geq \sigma_k$, and v is a unit vector, σ constitute the lengths of the semi-axes of the ellipsoid onto which the unit sphere is mapped. In Figure 1, for instance, $\sigma_1 u_1$ has u_1 a unit vector in the direction of one of the semi-axes with length σ_1. Given vector v_i, one can acknowledge the following:

$$Av_i = \sigma_i v_i, \ i = 1, 2, ..., k$$

The unit vectors $[v_1, ...v_k]$ are called the right singular vectors and they are orthogonal. The unit vectors $[v_1, ...v_k]$ are called the left singular vectors and are also orthogonal. Allowing V to be the matrix $[v_1, ...v_n]$ and U to be the matrix $[v_1, ...v_n]$, one can establish the following:

$$AV = [Av_1, Av_2, ..., Av_n]$$

$$= [\sigma_1 v_1, \sigma_2 v_2, ...\sigma_n v_n]$$

$$= \begin{bmatrix} \mathrm{u}_1 & \mathrm{u}_2 & \mathrm{u}_n \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \sigma_n \end{bmatrix} = U\Sigma$$

The matrix Σ is a diagonal matrix consisting of σ_n values in descending order. To complete the matrix factorization, since the columns of V are orthogonal, $VV^T = I$, one can multiply both sides by V_T, which yields the matrix factorization:

$$A = U\Sigma V^T$$

This leaves the left singular values as U, and the right singular values as the rows of V^T, completing the SVD. Rewriting matrix A as a linear combination of its constituent matrices yields:

$$A = U\Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T ... + \sigma_k u_k v_k^T$$

SVD can be used for up to as many terms as desired, until the original matrix is rebuilt. This application is extremely useful in the context of image compression, which will be further explained in Section 4.

II. Applying SVD

To solidify the matrix factorization process of SVD, one can review the following example of a matrix breakdown with real values on a step-by-step basis. Considering matrix A:

$$A = \begin{bmatrix} 3 & 2 \\ -2 & -3 \end{bmatrix}$$

Recall that the matrix factorization of A breaks down the matrix into three submatrices: U, Σ, and V^T. To begin, one must start with finding the left singular vectors: matrix U. In order to find this matrix, begin by finding AA^T and calling the resulting matrix B.

$$AA^T = \begin{bmatrix} 3 & 2 \\ -2 & -3 \end{bmatrix} \begin{bmatrix} 3 & -2 \\ 2 & -3 \end{bmatrix}$$

$$= \begin{bmatrix} 13 & -12 \\ -12 & 13 \end{bmatrix} = B$$

Next, one must find the eigenvalues of B to find the eigenvectors associated with the eigenvalues. One starts by finding the determinant of matrix B equal to zero and solve for λ.

$$det(B - \lambda I) = 0$$

$$det(\begin{bmatrix} 13 & -12 \\ -12 & 13 \end{bmatrix} - \begin{bmatrix} -\lambda & 0 \\ 0 & -\lambda \end{bmatrix}) = 0$$

After finding the determinant of the form $ad - bc$, one can factor the resulting polynomial equation to find that λ = 1, 25. These eigenvalues should be the same as that for the previous matrix. After finding the eigenvalues of matrix B, one can find the eigenvectors associated with these eigenvalues. The setup for computing these eigenvectors is as follows:

$$\begin{bmatrix} 13-\lambda & -12 \\ -12 & 13-\lambda \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

Inserting each eigenvalue into the equation above, and solving the resulting matrix multiplication as a system of linear equations yields the following eigenvectors:

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

These eigenvectors will constitute matrix U sought in the factorization, but before proceeding one must make sure that these vectors are $orthonormal$. The Gram-Schmidt Process makes the matrices orthogonal, if they are not already. Matrix B, being a symmetrical matrix, is already orthogonal, as the eigenvectors of real symmetric matrices are known to be orthogonal. The final step of this process, then, is to normalize these eignevectors, giving them a magnitude of 1, resulting in the final form of vector U to be used in the factorization:

$$\begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

The factorization for vector U can be used in finding vector V^T. To begin finding the values of vector V^T, one takes matrix A and finds A^TA resulting in matrix C:

$$A^TA = \begin{bmatrix} 3 & -2 \\ 2 & -3 \end{bmatrix} \begin{bmatrix} 3 & 2 \\ -2 & -3 \end{bmatrix}$$

$$= \begin{bmatrix} 13 & 12 \\ 12 & 13 \end{bmatrix} = C$$

Next, one must find the eigenvalue associated with matrix C to then find the eigenvectors. To begin, one finds the determinant of matrix C equal to zero and solve for λ.

$$det(C - \lambda I) = 0$$

$$det(\begin{bmatrix} 13 & 12 \\ 12 & 13 \end{bmatrix} - \begin{bmatrix} -\lambda & 0 \\ 0 & -\lambda \end{bmatrix}) = 0$$

After finding the determinant in the form of $ad - bc$, one can factor the resulting polynomial equation to find the eigenvalues to be λ = 1, 25. The eigenvalues are the same due to vector U and vector V are both transformation vectors composed from using the same matrices. After finding the eigenvalues of matrix C, one can find the eigenvectors associated with these eigenvalues. The setup for computing these eigenvectors is as follows:

$$\begin{bmatrix} 13-\lambda & -12 \\ -12 & 13-\lambda \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

Inserting each eigenvalue into the equation above, and solving the resulting matrix multiplication as a system of linear equations yields the following eigenvectors:

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

These eigenvectors will constitute matrix V matrix V fount in the factorization, but before proceeding, one must make sure that these vectors are also orthonormal. Matrix C, being a symmetrical matrix, is already orthogonal, as the eigenvectors of real symmetric matrices are known to be orthogonal. The next step of this process, then, is to normalize these eignevectors, giving them a magnitude of 1, resulting in the final form of vector V to be used in the factorization:

$$V = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

For the final step, one must find the transpose of vector V. The transpose of a vector is found by turning the columns into rows and laying them in order of the columns. The resulting V^T matrix is as follows:

$$V^T = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

The last matrix needed is Σ, a scalar matrix . Σ is found by taking the eigenvalues from finding the determinate earlier finding the positive square roots of the eigenvalues. The roots of the eigenvalues are placed diagonally in descending order with the rest of the matrix being zeroes. The resulting matrix is as follows:

$$\Sigma = \begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix}$$

Now, one has decomposed matrix A into rotational matrix U, scaling matrix Σ, and another rotational matrix V^T.

III. Gram Schmidt Process

The Gram Schmidt Process allows for the factorization of non-orthogonal matrices for SVD by making them orthogonal.

The Gram Schmidt Process begins by taking the span of the non-orthogonal matrices. Consider vectors X and Y:

$$X = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}$$

$$Y = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

Now one must take the span of the vectors X and Y equal to the subspace R. R are the orthogonal matrices needed for factorization in SVD.

R = span ($\begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$)

Now considering vector X equals the span of vector X, one can find an orthonormal basis for vector X. One must find the length of vector X by square rooting the vector and multiplying it by vector X. Considering the orthonormal basis is U, one knows the span of U if the same as the span of X.

$$X = span(X)$$

$$\sqrt{1+1+0} = \sqrt{2}$$

$$U = \sqrt{2}\begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}$$

Now one wants to find the orthogonal basis for vector X and Y, so one must find the span of U,Y. The orthonomal vectors of X and Y is equal to the span of X,Y and the span of U,Y. The span of U,Y is the same as the span of U,V. V is the orthonormal basis for vector Y. V is equal to Y - projection of Y onto X:

$$span(X,Y) = span(U,Y) = span(U,V)$$

$$V = Y - proj(x)Y$$

Next, one simplifies for V to find the orthonormal basis for vector Y.

$$V = \sqrt{\frac{2}{3}}\begin{bmatrix} \frac{-1}{2} \\ \frac{-1}{2} \\ 1 \end{bmatrix}$$

Vectors U and V are both orthonormal from each other with a length of 1 and they span R. This makes them ready to be factorized for SVD use.

IV. Contextualization In Image Compression

Having completed the SVD process for a 2x2 matrix, one can also apply the same methodology to images. Taking an image to be an n-by-m matrix (with an n-by-m byte size), one can perform the SVD process on the image. Recall from the introductory section, that matrix A can be written as the following linear combination:

$$A = U\Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T ... + \sigma_k u_k v_k^T$$

The factorization of A can be completed for k amount of terms, with each term progressively helping to complete the original matrix, but each term doing so less and less as k increases, due to the positioning of the larger eigenvalues on the Σ matrix first. In the context of image compression, each term in the linear combination form of A helps to rebuild the original image matrix, where the terms at the beginning rebuild

Fig. 2: A high-quality original image.

more of the matrix than their subsequent terms. For instance, consider the following image:

This image is 1578 x 833 pixels large, taking up around 1.3 megabytes of storage space. By applying SVD to this image, one can significantly reduce the amount of storage space required for the image. While the SVD process subtracts some visual detail, it would be nearly impossible to tell for an image of that size.

Fig. 3: A low-quality version.

With only a k value of 10 (completing the SVD for 10 terms), Figure 3 already has distinguishes similarities with the original image. In contrast to the original, however, this image only takes up 24 kilobytes of space.

Fig. 4: A compressed version of Figure 2.

This third rendered image, Figure 4 has a k value of 100. In figuring out the required storage size for an image, one can consider the following equation: $k(n+m+1)$. Each n and m term is representative of the size of the columns of matrix U, and rows of matrix V^T, and the σ value in the term is represented by the 1, as it is a scalar value. These three things constitute a single term, and the k serves as a multiplier for however many terms one wishes to continue the decomposition with. Figure 4, for instance, is a compressed rendering of a 1578 x 833 pixel image. These dimensions constitute n and m. The number of k terms used in the compression of 4 is 100. Therefore, the size of the compressed image is 241 kilobytes, or about 20% the size of the original image. Nifty, indeed!

V. Conclusion

Singular Value Decomposition is the math behind getting Netflix recommendations fingerprint recognition, and facial recognition. SVD is also involved in image compression. SVD in image compression is capable of creating an image of equal quality but as a smaller file size. This is capable due to the math involved in SVD. The image is turned into matrix and is decomposed into a rotational matrix, a scaling matrix, and another rotational matrix. This results in the image becoming layers where more layers are combined until the image looks like the original despite being a smaller file size. The illusion comes from our eyes being unable to distinguish the different from the decompressed image and the original as our eyes are not as detailed.

VI. References

James, David, Michael Lachance, and Joan Remski. ”Singular Vectorsâ Subtle Secrets.” The College Mathematics Journal 42.2 (2011): 86. Web. 1 May 2017.

Image Compression Student Paper

Khan, Sal. ”The Gram-Schmidt Process.” Khan Academy. Khan Academy, n.d. Web. 01 May 2017.