Data Analysis and Matrix Decomposition

Tristan Gaeta

University of Puget Sound

May 2, 2021

Definitions of matrix Multiplication

Suppose A is an $m \times n$ matrix with columns $\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_n$ and B is a $\mu \times n$ matrix with columns $\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_n$. We can define the matrix product AB^* as

$$AB^* = \sum_{k=1}^{n} \boldsymbol{a}_k \boldsymbol{b}_k^*$$

We can express any matrix product as summation of outer products.

The Singular Value Decomposition

Suppose X is an $m \times n$ matrix with singular values $\sigma_1, \sigma_2, \cdots, \sigma_n$. Matrix U is unitary with columns $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_m$ that span the column-space of X. Matrix V is unitary with columns $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n$ that span the row-space of X. Matrix S is a diagonal matrix with singular values of X in decreasing order. There exists a decomposition such that

$$X = USV^*$$

Using our previous definition we can express the matrix X as

$$X = \sum_{k=1}^{n} \sigma_k \boldsymbol{u}_k {\boldsymbol{v}_k}^*$$

Because of the decreasing order of singular values, we can approximate matrix X using values.

Example of Data Compression

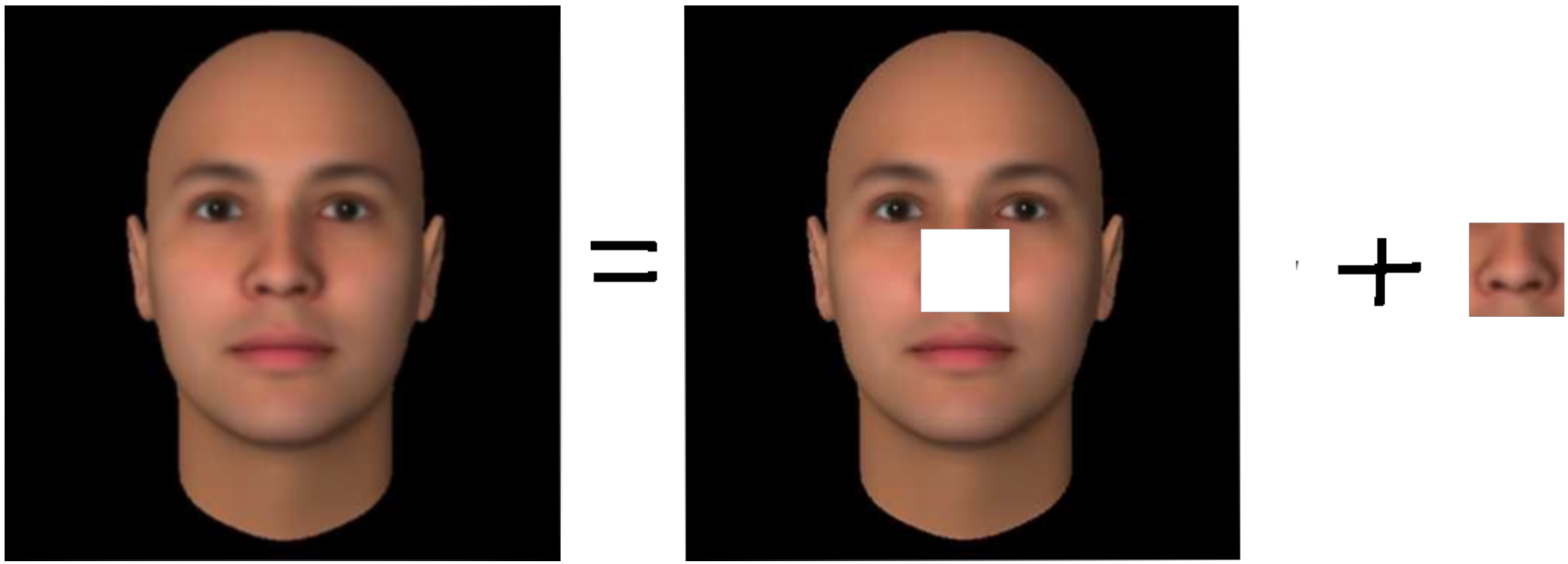
Suppose we have a set of several thousand high definition images of peoples' faces. We want to construct a new face as a linear combination of our other faces.

Instead of recording each individual pixel to represent a face, we can find clusters of pixels that are strongly correlated throughout the images, and use one value for their color. Doing this process, via the SVD, will reduce the dimensionality of the vector needed represent a face.

How, then, could we control the characteristics of the face we are designing?

Example of Data Compression

By breaking our images up into their different characteristics, we can create a span of different noses, mouths, eyes, etc. to choose from when designing our face.



Definitions of matrix Multiplication

Suppose A is an $n \times m$ matrix with columns $\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_m$ and B is a $n \times \mu$ matrix with columns $\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_\mu$. We can define any entry at row i and column j of the matrix product A^*B as

$$\begin{aligned}
[A^*B]_{ij} &= \sum_{k=1}^{n} [A^*]_{ik}[B]_{kj} \\
&= \sum_{k=1}^{n} \overline{[\boldsymbol{a}_i]_k}[\boldsymbol{b}_j]_k \\
&= \langle \boldsymbol{a}_i, \boldsymbol{b}_j \rangle
\end{aligned}$$

We can express any entry of a matrix product as an inner product of two vectors.

Key Aspects of the SVD

Suppose X is an $m \times n$ matrix with columns $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n$. We can define any entry at row i and column j of the correlation matrix X^*X as

$$[X^*X]_{ij} = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$$

In other words each entry of the matrix, $[X^*X]_{ij}$, is the Hermitian inner product of the ith and the jth columns of the original matrix, X. Similarly the entries of the correlation matrix XX^* are inner products of two rows of matrix X.

The Netflix Prize

In 2006 the movie rental service Netflix, before creating their online streaming service, hosted a competition where, for the prize of one million dollars, contestants were asked to improve Netflix's current recommender system, Cinematch, by 10% or more. The participants were given a data set that contained the sparse ratings from 480,189 user accounts on 17,770 different movie tittles. The set only contained 100,480,507 ratings total, each also include the date the rating was given.

The Netflix Prize

Lets construct the $m \times n$ data matrix, X, such that each column, $\boldsymbol{x}_j$, contains all of the ratings for one movie in the data set, in some order. The end goal is to fill in the unknown values of the data matrix X using correlations within users' movie preferences.

We can assume there exists a SVD for our data matrix, so if we would like to see what person i would rate item j we can see by the definition of matrix multiplication that

$$\begin{aligned}
[X]_{ij} &= \sum_{k=1}^{m} [US]_{ik} [V^*]_{kj} \\
&= \sum_{k=1}^{m} [US]_{ik} \overline{[V]}_{jk} \\
&= \langle \widehat{\boldsymbol{v}}_j, \widehat{\boldsymbol{u}}_i \rangle \qquad \widehat{\boldsymbol{v}}_j \in \mathcal{R}(V),\ \widehat{\boldsymbol{u}}_i \in \mathcal{R}(U)
\end{aligned}$$

The Netflix Prize

Our goal is to approximate the vectors $\widehat{\boldsymbol{v}}_j$ and $\widehat{\boldsymbol{u}}_i$ for every unknown value, while keeping the known entries as close to their actual value as possible. We can achieve this by minimizing the system

$$\min_{\widehat{X}} \sum_{k_{ij} \in K} (k_{ij} - \langle \widehat{\boldsymbol{v}}_j, \widehat{\boldsymbol{u}}_i \rangle)^2 + \lambda(\|\widehat{\boldsymbol{v}}_j\|^2 + \|\widehat{\boldsymbol{u}}_j\|^2)$$

We can approximate a solution to this system by using Stochastic Gradient Descent or Alternating Least Squares. We can improve our model using biases and different data points provided in the Netflix data set.