Dimensionality reduction and noise removal of face images with Non-Negative Matrix Factorization

TMA4320 Industrial Mathematics project.

Martin Ludvigsen

NTNU - Institutt for Matematiske fag

Introduction

”Mo’ data, Mo’ problems”

Question: How can we learn underlying structures from large amounts of data?

Answer: Dimensionality Reduction.

Some examples of dimensionality reduction

- IQ as (bad) 1D measure of intelligence.
- Political compass as (also bad) 2D measure of political spectrum.
- Myers Briggs/Big five test as 4D/5D measures of personality type.

Reality more complex

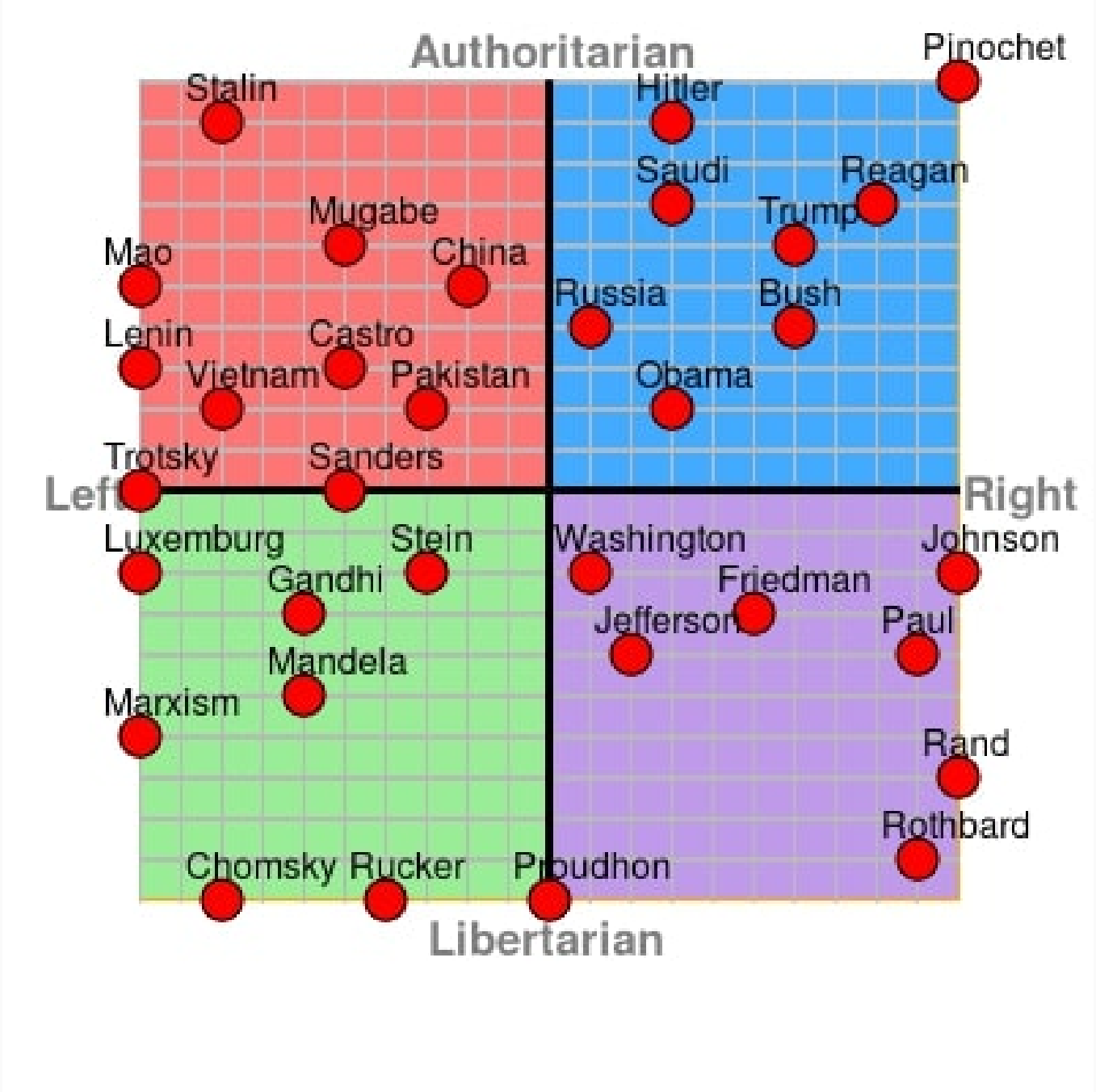


Figure 1: Political compass (with some errors, but you get the point).

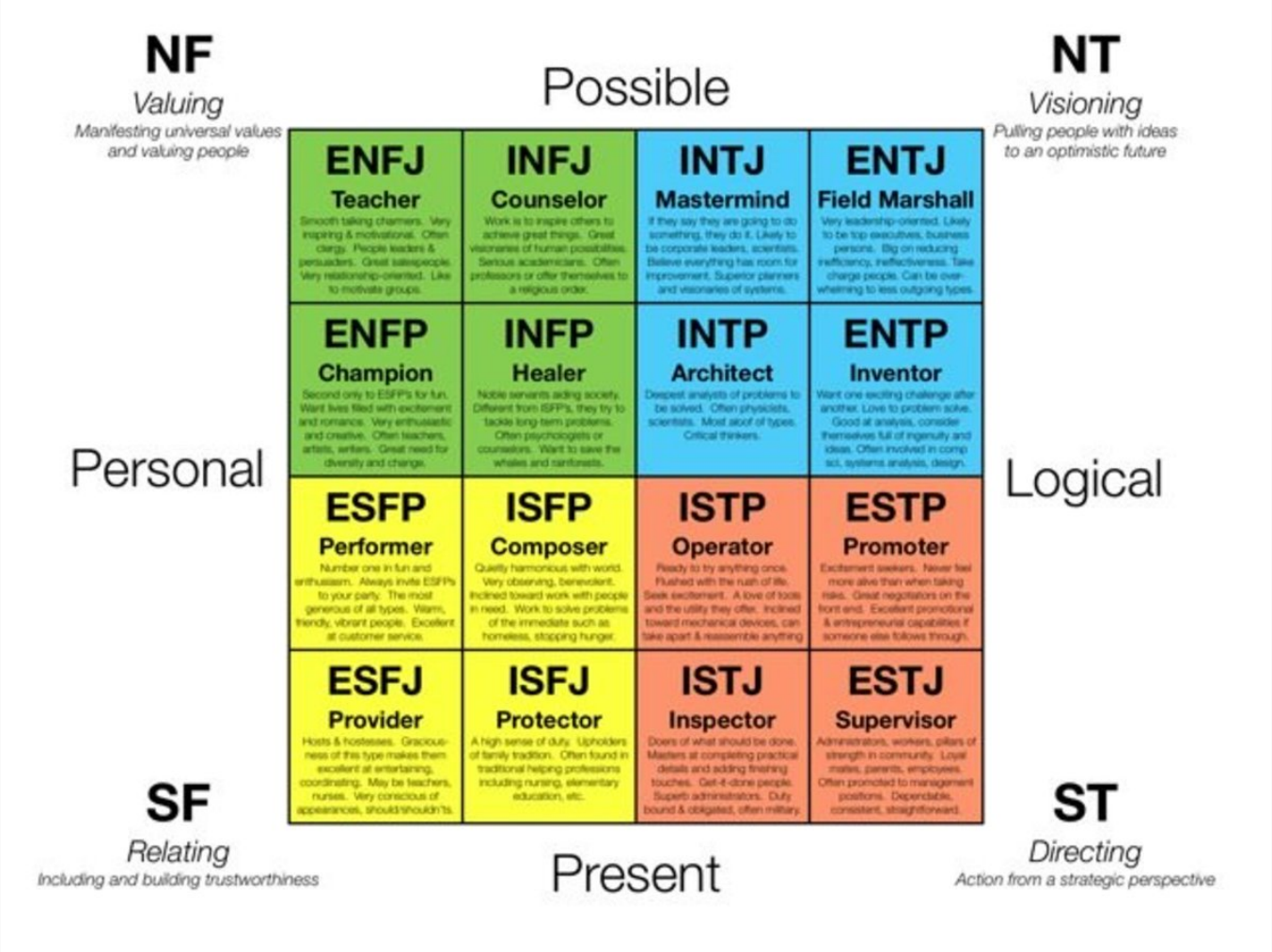
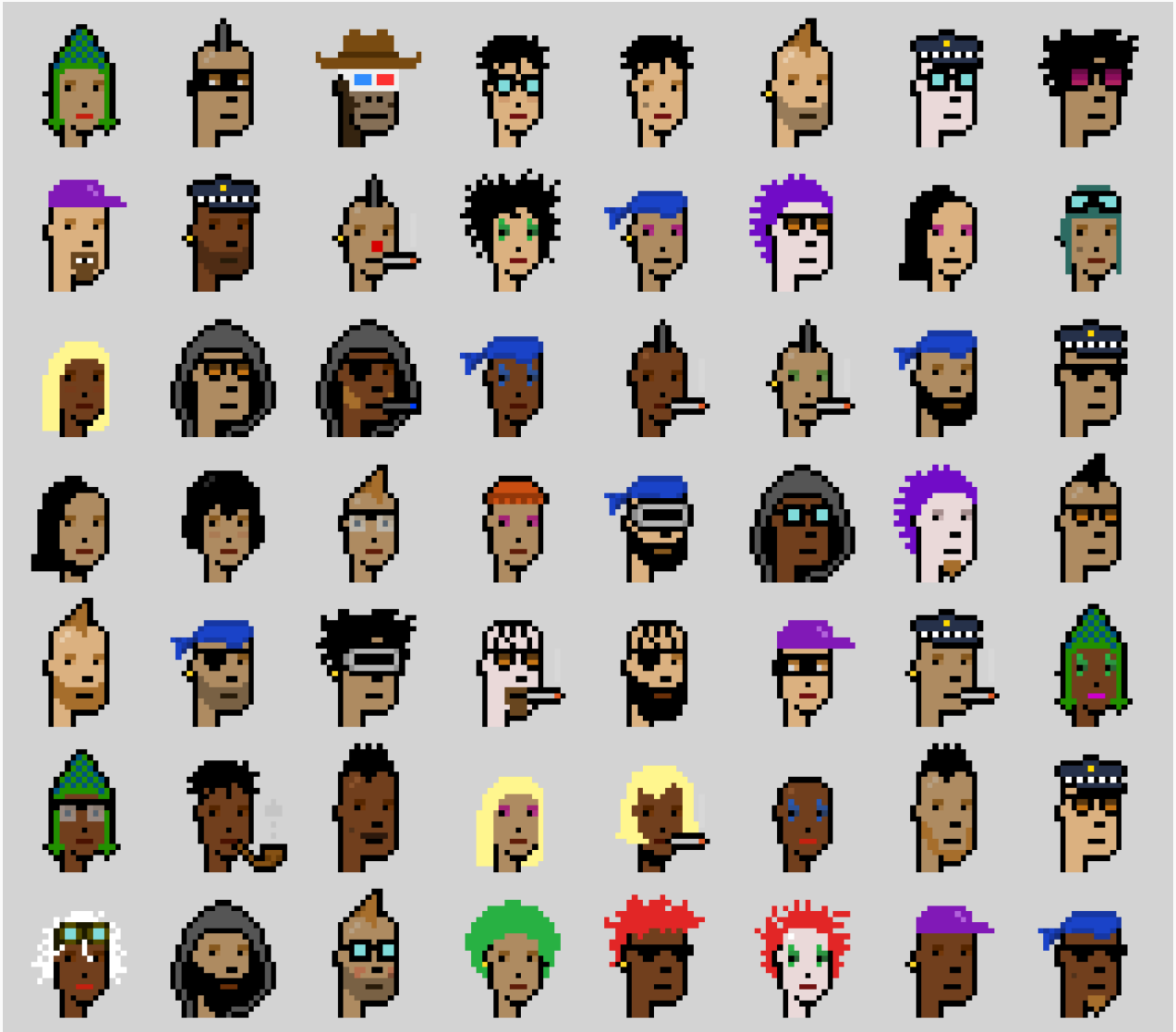


Figure 2: Myers Briggs categorizes personalities along 4 axis, resulting in 16 personality types .

Dimensionality reduction in images

Imagine you have thousands of face images. Can you understand these images in only a few hundred images? A few tens of images?

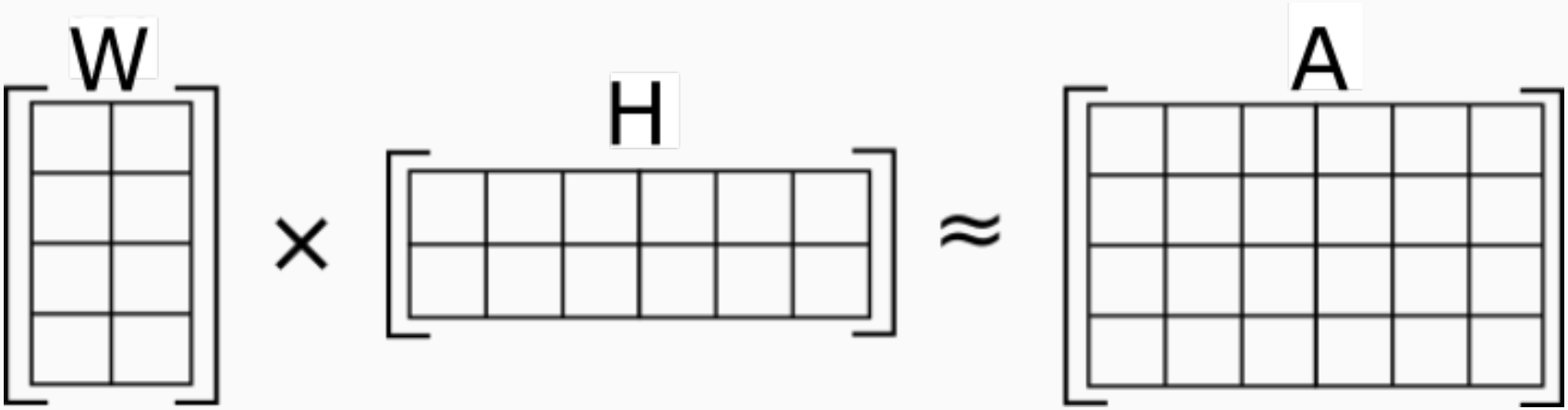


Non-Negative Matrix Factorization (NMF)

The NMF of a non-negative $m \times n$ matrix A is the matrix product

$$A \approx A_{\text{NMF}} = WH, \tag{1}$$

where W is a $m \times d$ non-negative matrix and H is a $d \times n$ non-negative matrix with $d \ll m, n$.



NMF as change of basis

Each column a_i of A can be described as a **non-negative linear combination** of the columns of W.

$$a_i \approx Wh_i = h_{i1}w_1 + h_{i2}w_2 + \dots \tag{2}$$

Call the columns of W **basis vectors** that should contain interesting information about the data.

Movie Example

Matrix of 6 different users rating of 6 different movies between 1 and 5. There are two ”genres” of movies, Marvel movies and Lord of the Rings movies. The users are mostly split between liking one genre or the other.

	User 1	User 2	User 3	User 4	User 5	User 6
Avengers: Infinity War	5	1	3	1	3	2
LotR: Fellowship of the Ring	2	4	1	4	1	4
Avengers: Endgame	4	1	5	1	4	2
LotR: Two Towers	1	4	2	5	2	4
Spiderman: Homecoming	3	1	5	3	3	2
LotR: Return of the King	2	4	2	4	2	4

Movie Example

Assume we are given a matrix of 6 different users rating of 6 different movies between 1 and 5. There are two ”genres” of movies, Marvel movies and Lord of the Rings movies. The users are mostly split between liking one genre or the other.

	User 1	User 2	User 3	User 4	User 5	User 6
Avengers : InfinityWar	5	1	3	1	3	2
LotR : FellowshipoftheRing	2	4	1	4	1	4
Avengers : Endgame	4	1	5	1	4	2
LotR : TwoTowers	1	4	2	5	2	4
Spiderman : Homecoming	3	1	5	3	3	2
LotR : ReturnoftheKing	2	4	2	4	2	4

Idea: Apply NMF with $d = 2$. Hopefully we can ”split” the data?

$$\approx \begin{matrix} \\ \\ \\ \\ \\ \\ \end{matrix} \begin{matrix} w_1 \\ \begin{bmatrix} 1.03 \\ 0.08 \\ 1.24 \\ 0.15 \\ 0.99 \\ 0.28 \end{bmatrix} \end{matrix} \begin{matrix} w_2 \\ \begin{bmatrix} 0.24 \\ 1.29 \\ 0.21 \\ 1.41 \\ 0.48 \\ 1.26 \end{bmatrix} \end{matrix} \begin{matrix} h_1 & h_2 & h_3 & h_4 & h_5 & h_6 \\ \begin{bmatrix} 3.43 & 0.11 & 3.74 & 0.55 & 2.84 & 1.04 \\ 0.77 & 2.98 & 0.83 & 3.27 & 0.88 & 2.87 \end{bmatrix} \end{matrix}$$

- Each column of W contains large values for one of the movie "genres" and small values of the other.
- We have split the data into two genres.
- Each column of H correspond to each user. Larger number $\rightarrow$ larger preference.
- $6 \times 6 = 36$ components $\rightarrow 2 \times 2 \times 6 = 24$ components.

Applications?

- Learning structures from large quantities of data.
- Learning underlying structures is also useful for applications like denoising.

Some theory

Eigenvalue decomposition

From linear algebra we know that for a square diagonalizable matrix A we have

$$A = PDP^{-1} \tag{3}$$

and if the matrix P is orthogonal ($P^TP = PP^T = I$)

$$A = PDP^T. \tag{4}$$

P contains the eigenvectors of A and D is diagonal containing the eigenvales of A.

Can write this similar to NMF with $W = P$ and $H = DP^T$

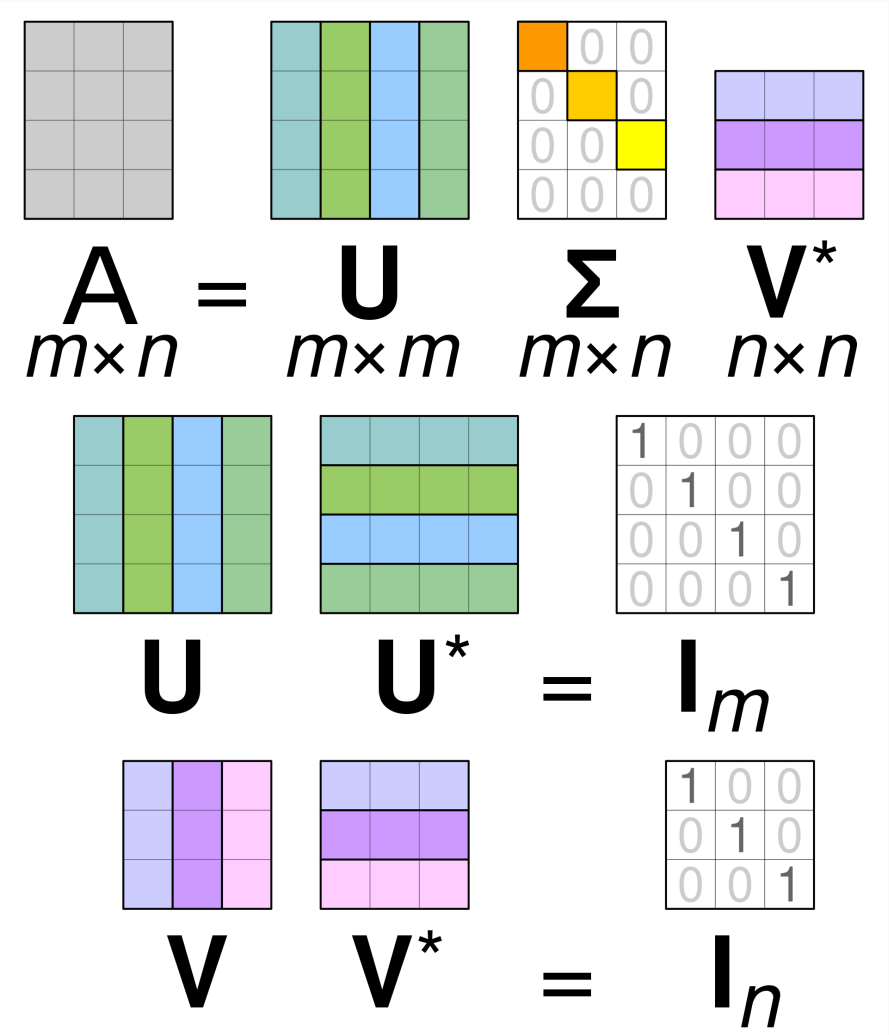
$$A = WH, \tag{5}$$

but this does in general not satisfy non-negativity.

Singular Value Decomposition (SVD)

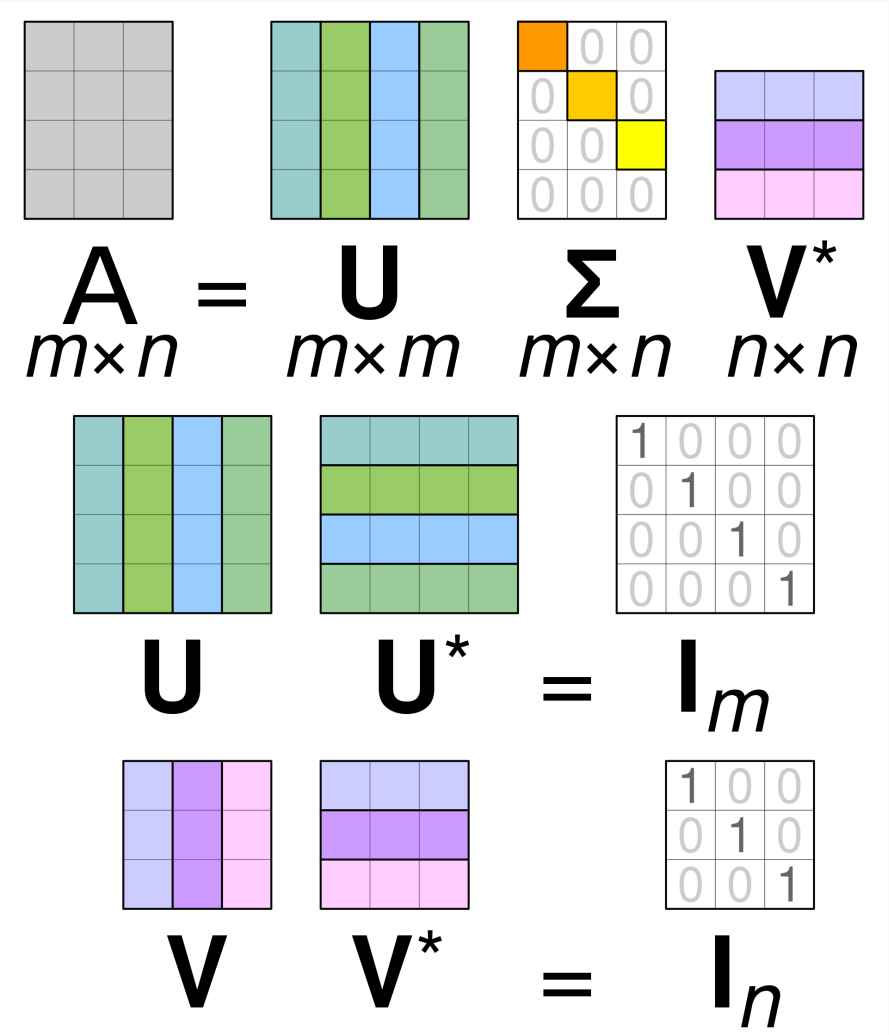
Eigenvalue decomposition does not exist for diagonal, and non-diagonalizable matrices, which leads us to a generalization, the SVD

$$A = U\Sigma V^T. \quad (6)$$



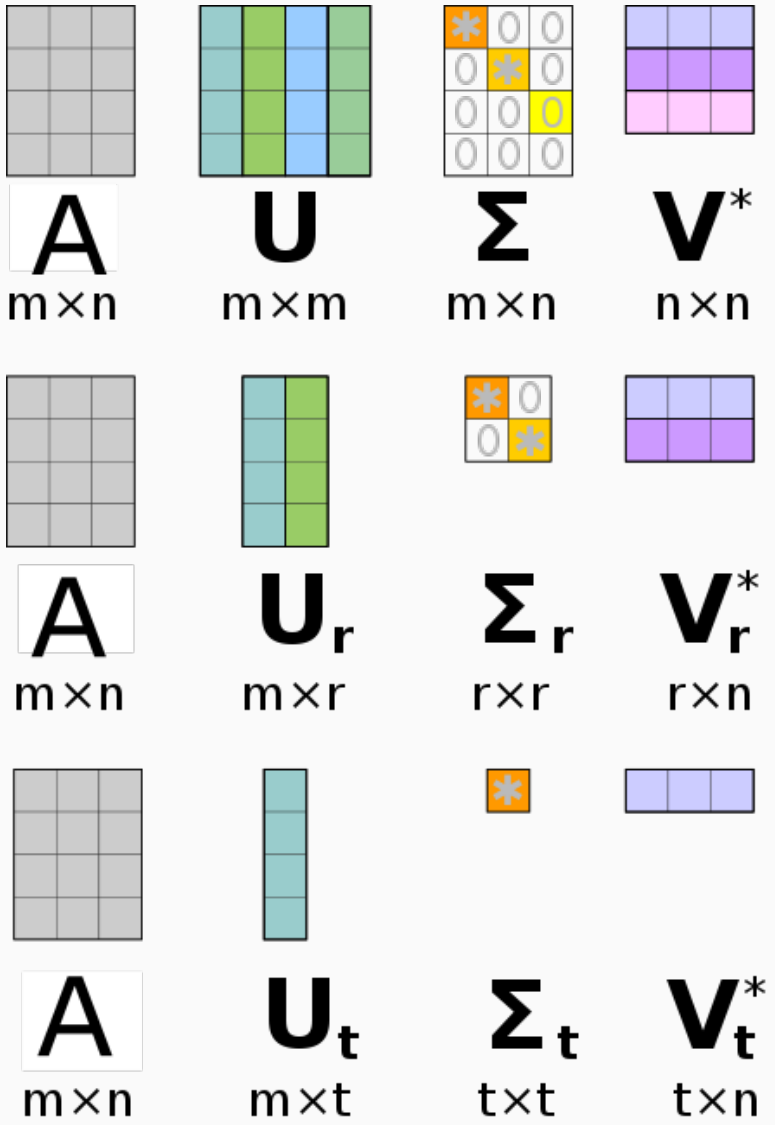
Singular Value Decomposition (SVD)

The SVD is equivalent to the eigenvaluedecomposition $A = PDP^T$ if A has non-negative eigenvalues.
The SVD **always** exists and is **unique** for all matrices A.



Reduced SVD

- Idea: Remove parts of the SVD corresponding to zero singular values (eigenvalues for square A) $\rightarrow$ perfect representation of A.
- Remove parts corresponding to small singular values, $\rightarrow$ good reconstruction of A.
- Dimensionality reduction.



SVD to NMF

Can also write SVD similar to the NMF

$$A = U\Sigma V^T = WH, \tag{7}$$

with for example $W = U$ and $H = \Sigma V^T$, which does not satisfy non-negativity.

Main takeaway: It makes sense to look for a factorization $A \approx WH$ where W and H are small. Smaller representations are easier to understand and compute.

Non-Negative Matrix Factorization

We can calculate the NMF by solving the problem

$$\min_{W,H} \|A - WH\|_F, \quad \text{so that } W, H \text{ is non-negative,} \tag{8}$$

where $\|.\|_F$ is the Frobenius norm. The SVD is essentially the solution to the same problem but without the non-negativity constraint.

This is a **constrained optimization** problem. Usually hard to solve.

Uniqueness

$$\min_{W,H} \|A - WH\|_F, \quad \text{so that } W, H \text{ is non-negative,} \tag{9}$$

The solution W and H to this problem is **non-unique**. There are infinitely many W and H that solve the problem (and in some cases, infinitely many WH). You will test this.

NMF in 1D

"NMF" in 1D, find two non-negative numbers w and h so that when you multiply them together you get a,

$$\min_{w,h} |a - wh|, \quad 0 \leq w, h. \tag{10}$$

Has infinitely many solutions $w = a/h$ and $h = a/w$.

Situation slightly more complicated in higher dimensions, but solution is similarily non-unique.

When to use NMF?

Question: What types of (non-negative) matrices A can be factorized well

$$A \approx WH, \tag{11}$$

and how does this depend on d?

Answer: We do not really know. Unclear what types of matrices can be well approximated with NMF. Should see better factorization as d increases.

SVD much less complex.

Why use NMF?

NMF is hard to solve, is non-unique and has little theory guaranteeing that it yields a good reconstruction. Why even bother?

- In many applications, the matrix A is non-negative (movie ratings and images), and we also want to interpret W and H as non-negative.
- The non-negativity gives the NMF gives it some interesting properties that are being researched. For example, it tends to create interesting features in the columns of W.
- NMF seems to be quite resilient to so-called overfitting, which will become relevant towards the end of the project.

Algorithm for solving the NMF

The algorithm we will be using is a relatively simple multiplicative update

$$(H_{k+1})_{ij} \leftarrow (H_k)_{ij} \cdot \frac{(W_k^T A)_{ij}}{(W_k^T W_k H_k)_{ij}}, \tag{12}$$

$$(W_{k+1})_{ij} \leftarrow (W_k)_{ij} \cdot \frac{(A H_{k+1}^T)_{ij}}{(W_k H_{k+1} H_{k+1}^T)_{ij}}, \tag{13}$$

where $(A)_{ij}$ denotes the component at the i-th row and j-th column of the matrix A, and k is an iteration index.

Relatively expensive algorithm, and convergence is quite slow.

Algorithm for solving the NMF

Can show that the iterations satisfy

$$\|A - W_{k+1}H_{k+1}\|_F \leq \|A - W_k H_k\|_F, \tag{14}$$

which you will test in the tasks.

However, the algorithm does NOT necessarily converge to a **global** minimizer, only a **local** one.

Algorithm for solving the NMF

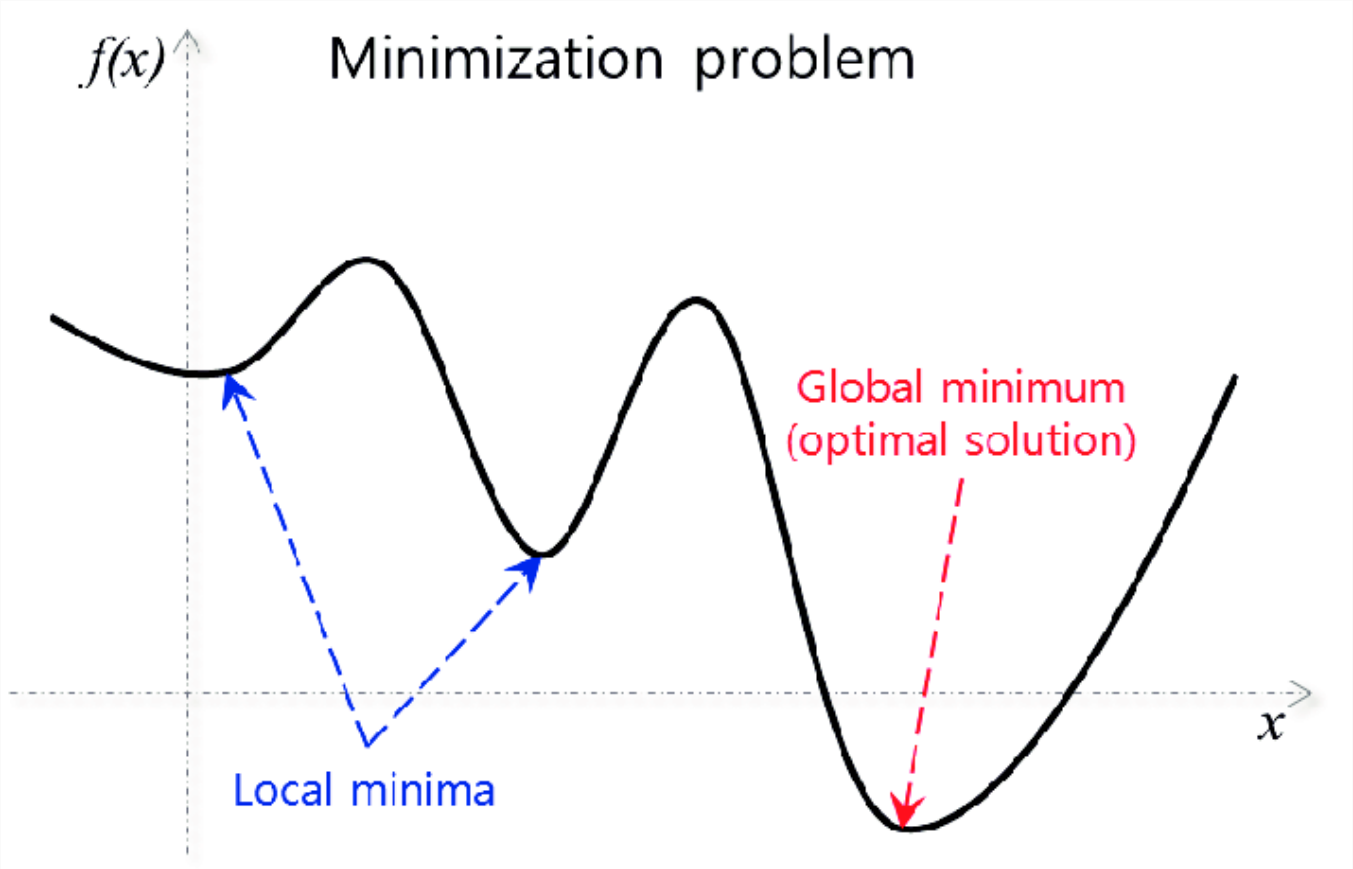


Figure 3: NMF algorithm might get stuck in these local minimizers, and will not find the true solution. Depends on where you start the iterations

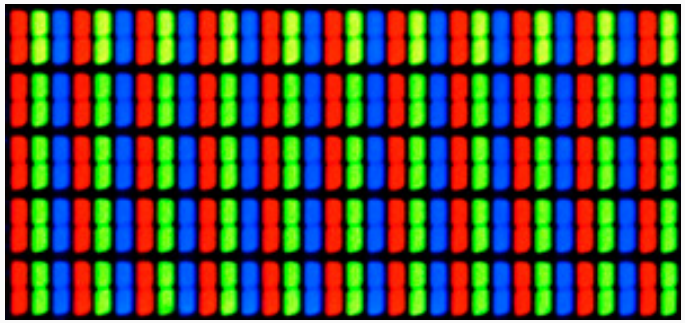
Summary so far

- Want to factorize $m \times n$ matrix $A \approx WH$ where W is $m \times d$ and H is $d \times n$ with $d \ll m, n$.
- Know from theory about eigenvalue decomposition and SVD that this makes sense to do.
- We saw how this could for example be used on a dataset of movie ratings to cluster the data and understand what users preferred what kinds of movies.
- Want non-negativity because it makes sense in many applications.
- We have a relatively simple algorithm for calculating the NMF...
- ... but it will not necessarily find an optimal NMF, and it is unclear wether or not the NMF will be able to produce a good factorization to begin with.
- Questions so far?

Images

A lot of science, technology and medicine is concerned with images. In this project we will investigate a particular dataset of ”face” images.

An RBG image consists of $m_x \times m_y$ pixels, with 3 color channels.

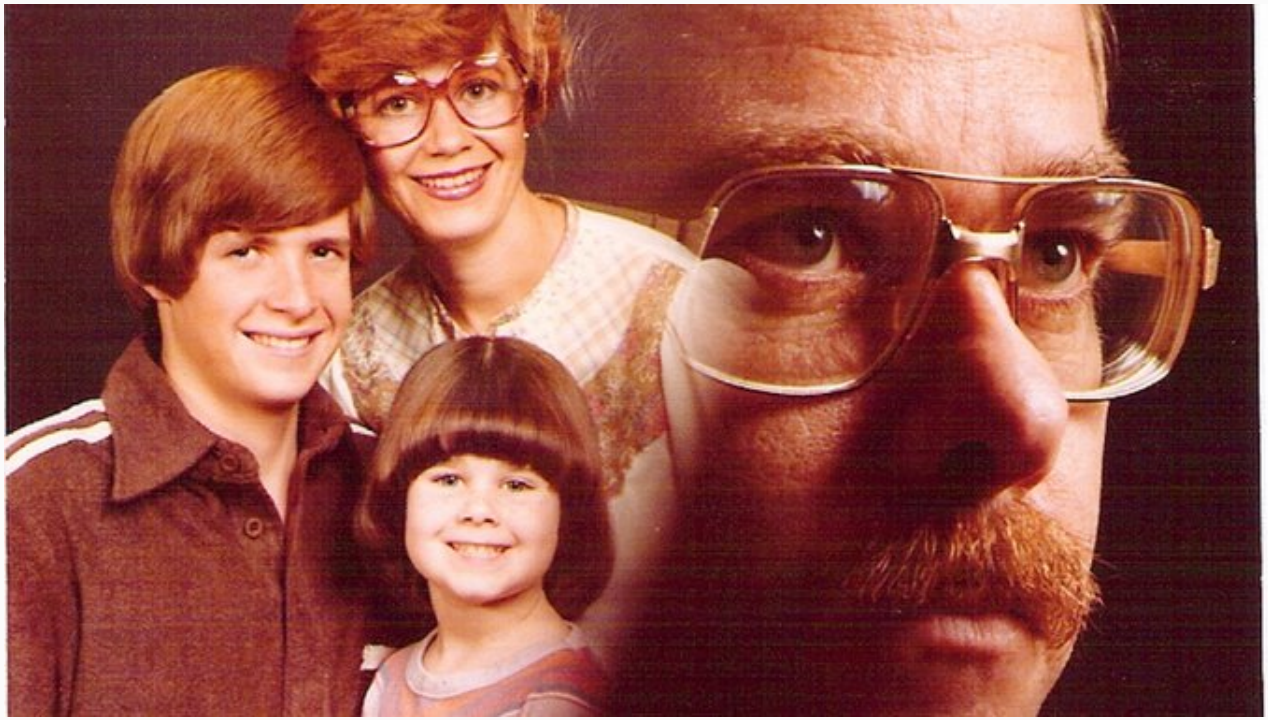


Datasets of images

- This means that an RBG image can be stored in a $m_x \times m_y \times 3$ NumPy array, and can be reshaped to a $3m_x m_y$ (vector) NumPy array.
- A dataset of N such images can then be stored in a non-negative (!) $3m_x m_y \times N$ (matrix) NumPy array, and we can apply the NMF to this matrix.
- In this case, each row corresponds to a certain color channel in a certain pixel and each column corresponds to an image.
- What will the columns of W, the ”hidden features” in the dataset look like?

Opacity

Opacity (opposite of transparency) stored in alpha channel of RGBA images.



Opacity

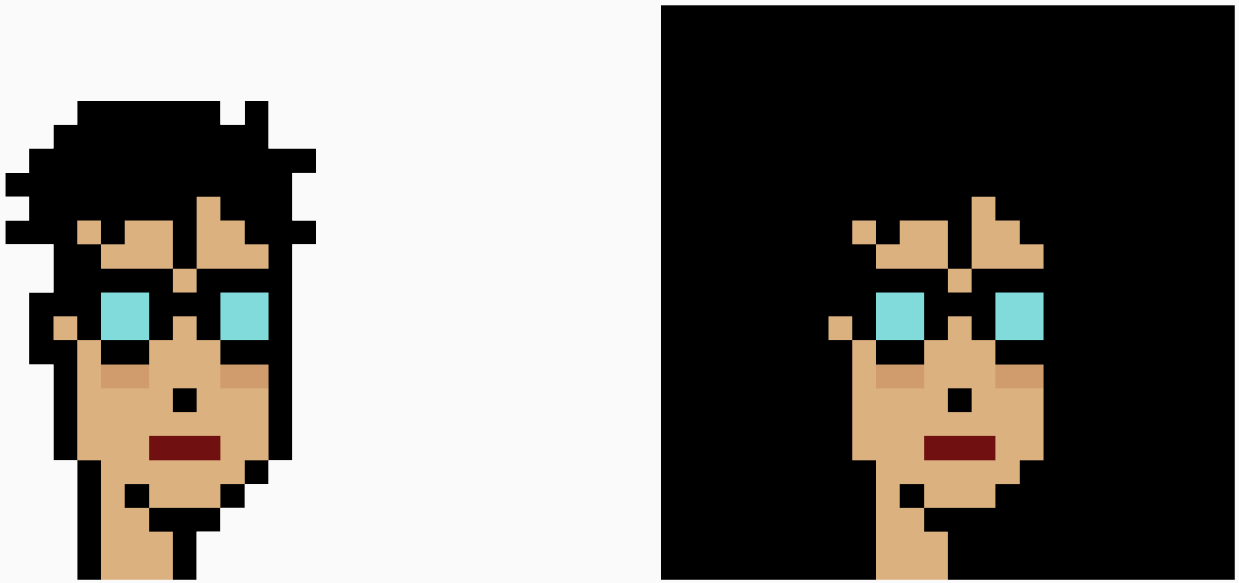
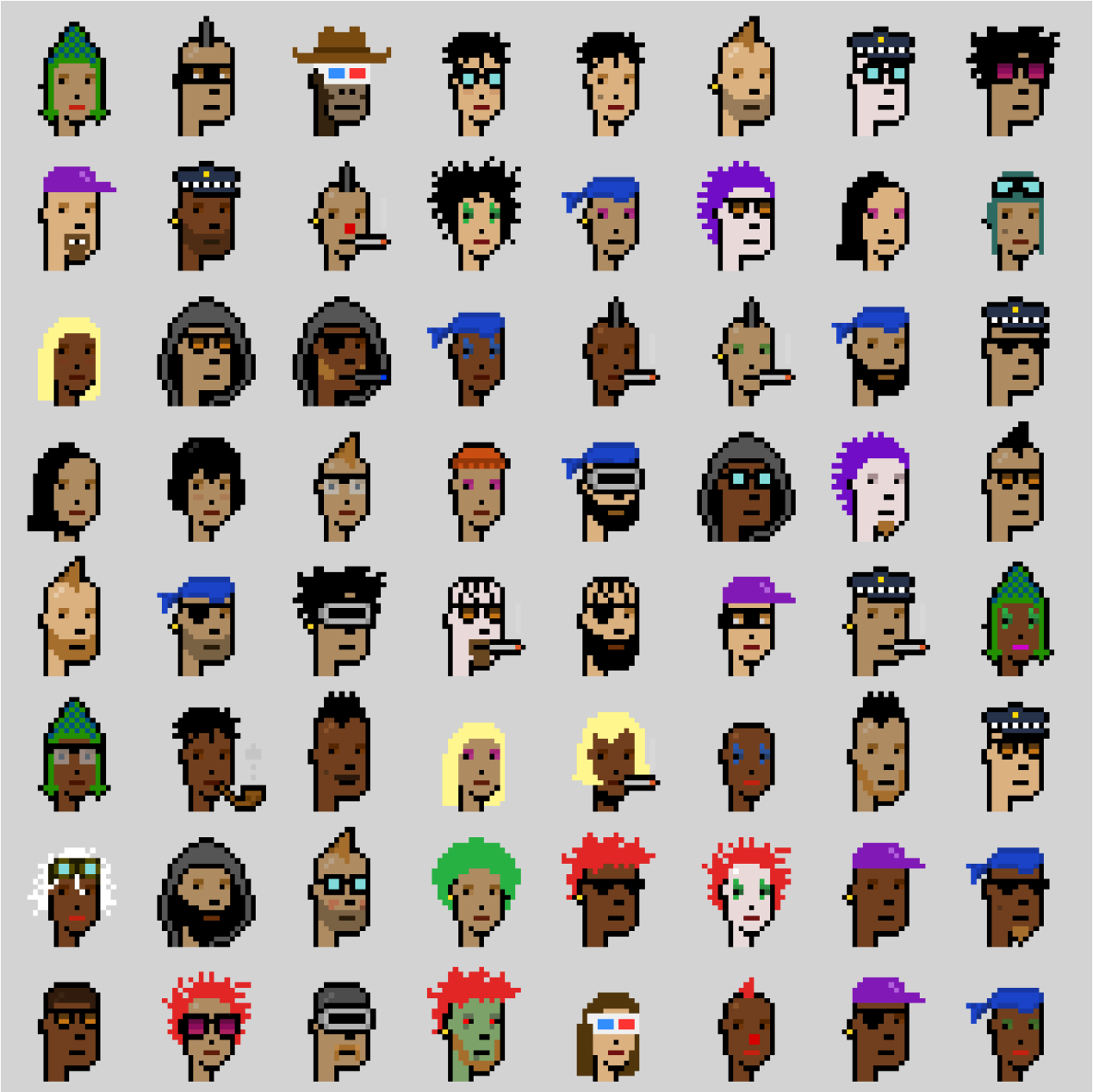


Figure 4: With and without opacity. Pixels are either completely opaque or completely transparent. Black pixels coincide with the background without opacity.

Cryptopunks



Basis vectors from NMF

Denoising

(Gaussian) Noise occurs naturally in images. Just try taking a dark image with your smartphone. We assume additive noise

$$A_{\text{noisy}} = A + \sigma E, \tag{15}$$

where each component of E contains independent realizations of a standard normal distribution, and $\sigma > 0$ is the so-called noise level.

We will only add noise in the color channels, and only in the non-zero ones.

Noise example

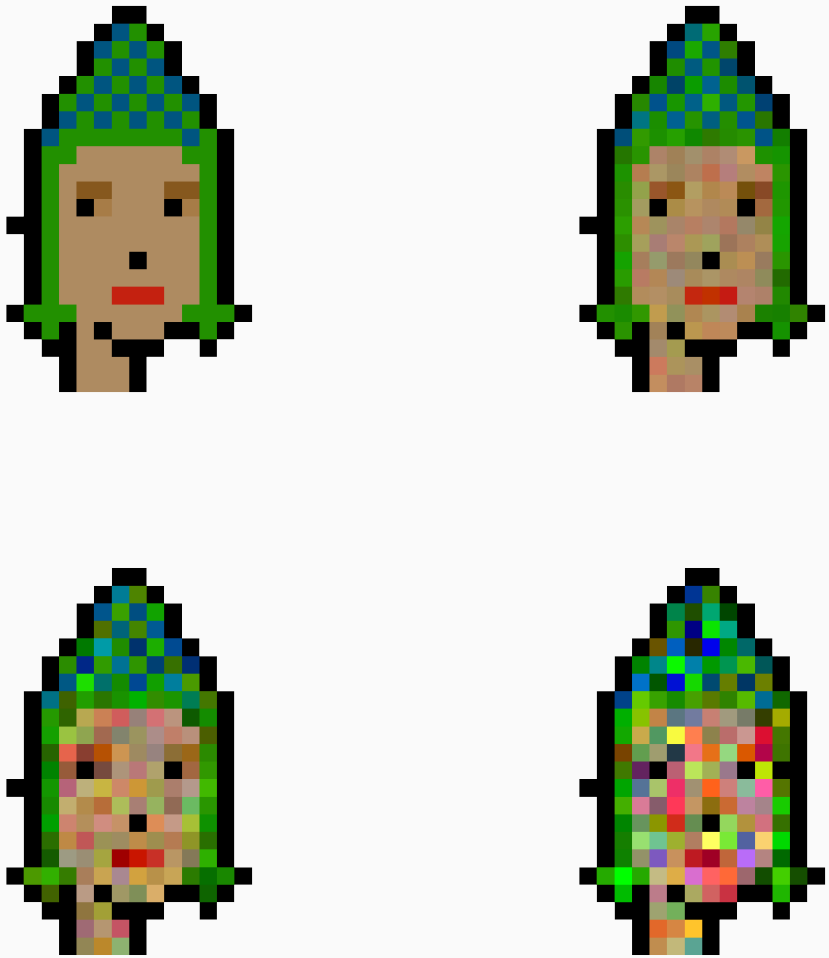


Figure 5: Noiseless image and images with added Gaussian noise with noise level $\sigma = 0.05, 0.1$ and 0.2 respectively.

Denoising with NMF

Noisy dataset + NMF: $A_{\text{noisy}} \approx WH \rightarrow WH$ contains denoised images!

How well this works depends heavily on the choice of d, and also the size of the dataset N:

- Low $d \rightarrow$ not enough basis vectors to learn the different structures of the data. Reconstructions will have little noise, but will have wrong skin color, hat/hair shapes etc.
- High $d \rightarrow$ too many basis vectors means that they will contain noise in order to fit the noisy data. Goal is to discard the noise!
- Large $N \rightarrow$ more data means better results. Can remove more noise.

Overfitting and underfitting

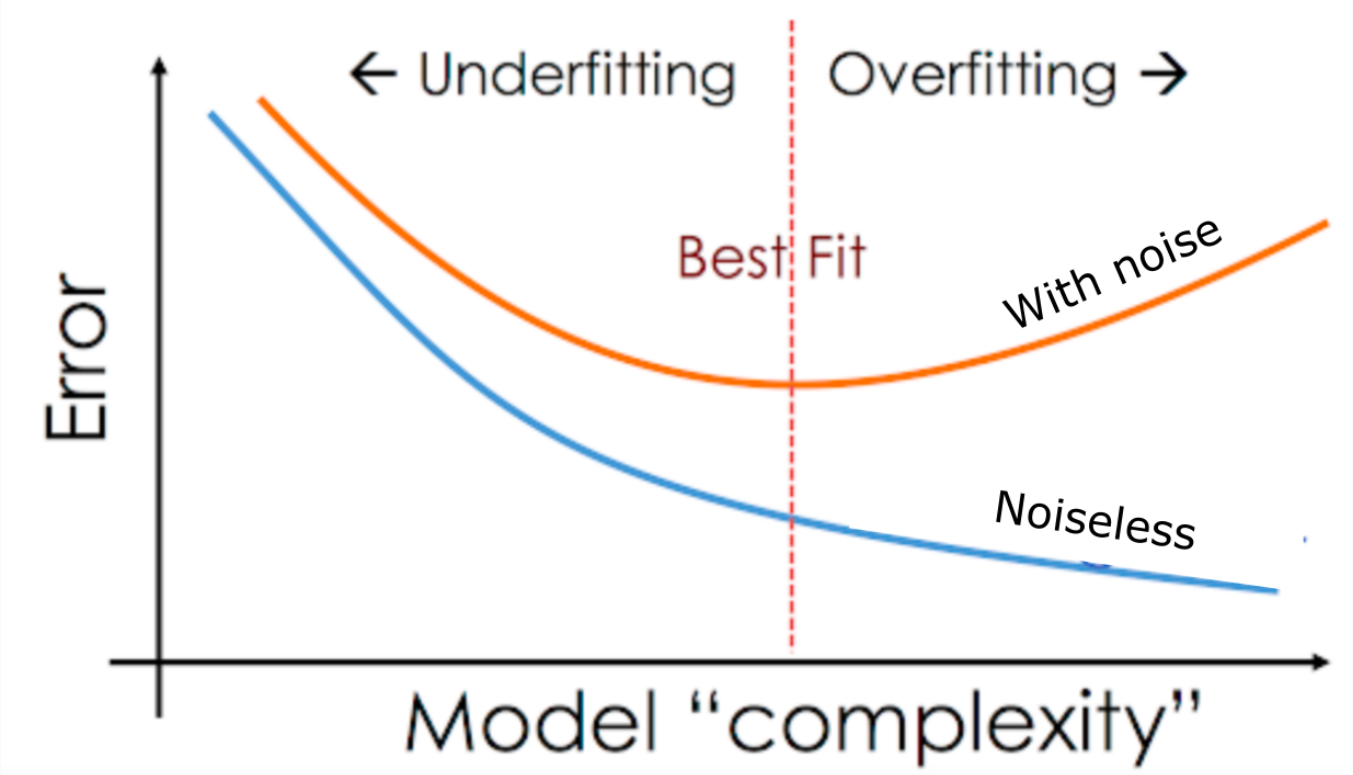


Figure 6: Over/underfittng is an extremely important and general concept in Machine Learning. Here "model complexity" refers to the value d, and error refers to the reconstruction error $||A - WH||_F$, where W and H are calculated with and without noise.

Handed out code

You will be handed out code containing

- Code to load the images to python (using opencv).
- Example code for plotting many images.
- Code for adding noise to images and plotting them.

Questions?