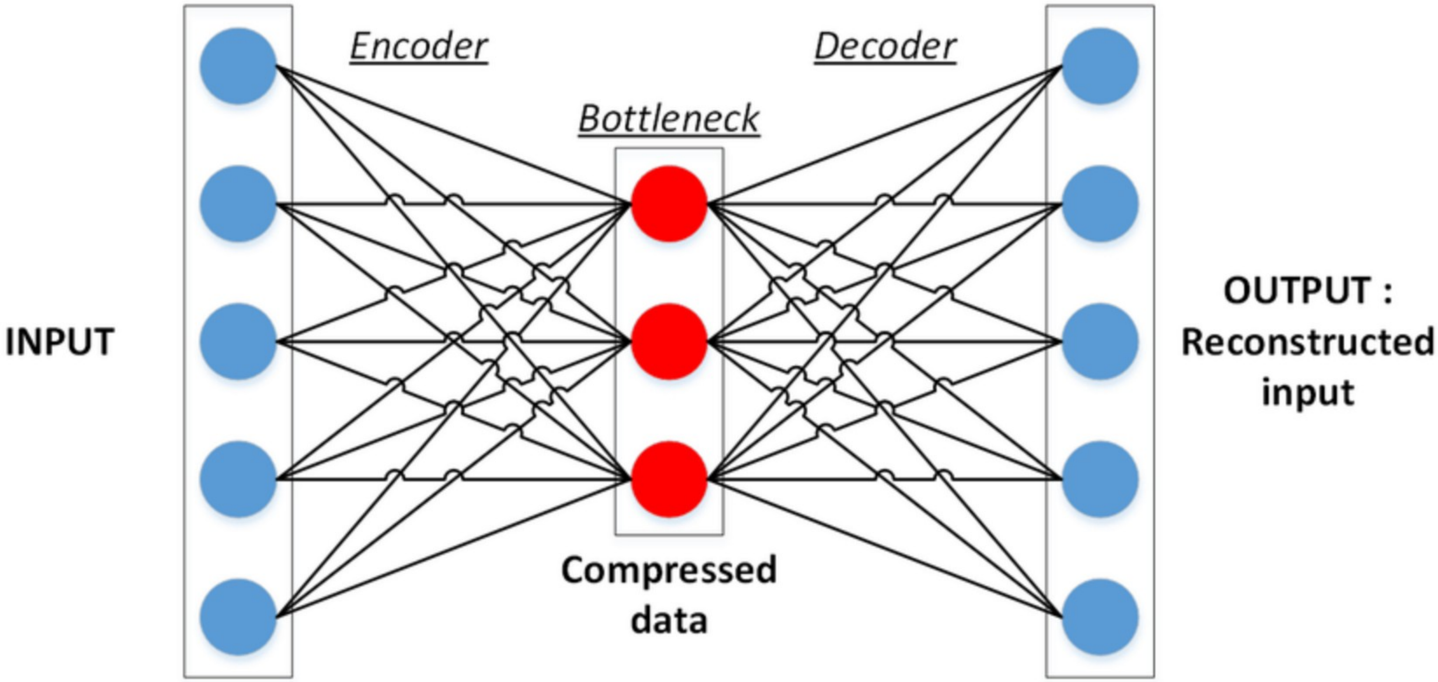
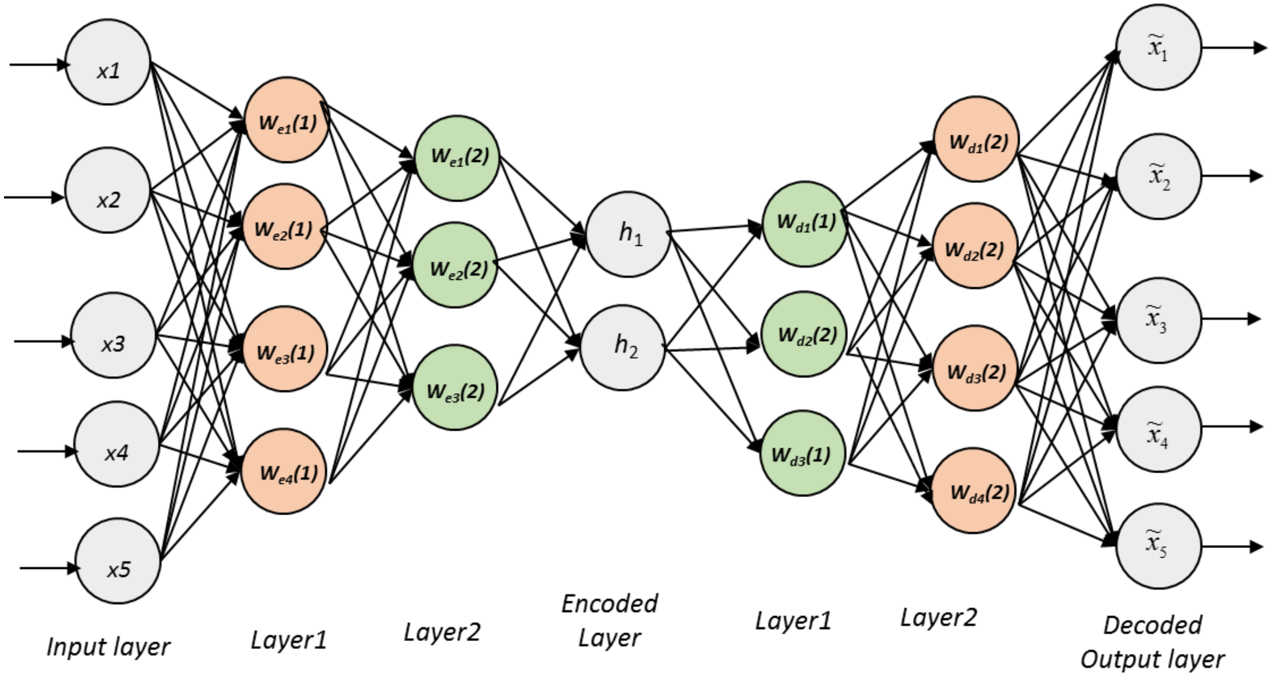
Autoencoders

Nathan Barnes

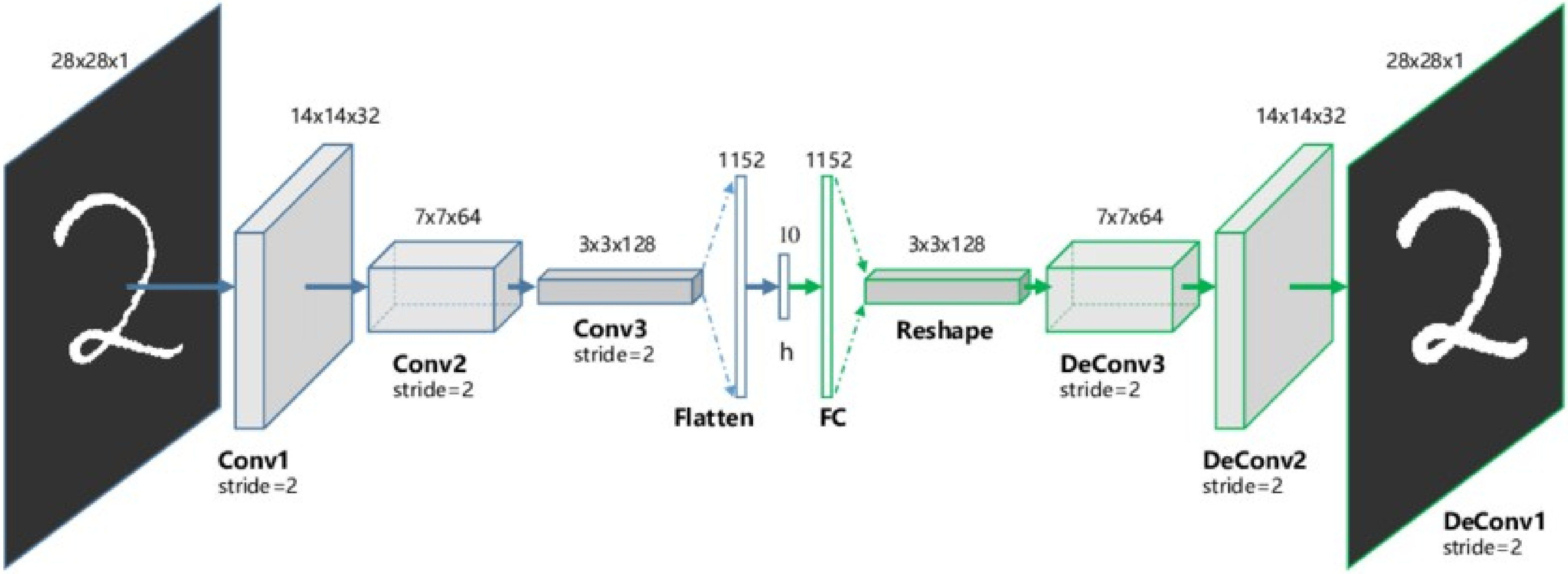
What is an autoencoder?



https://stackabuse.com/autoencoders-for-image-reconstruction-in-python-and-keras/



https://medium.com/@venkatakrishna.jonnalagadda/sparse-stacked-and-variational-autoencoder-efe5bfe73b64

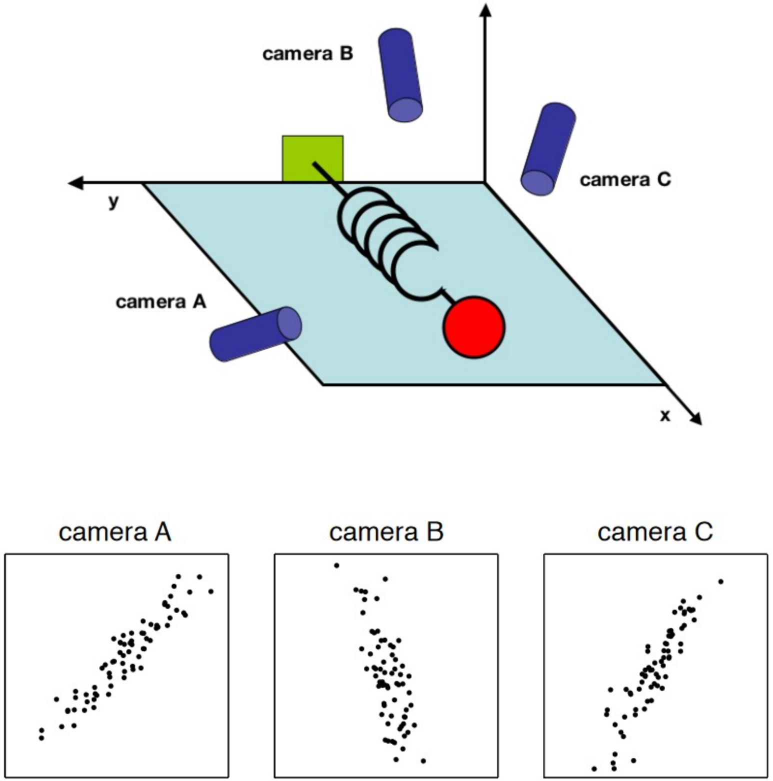


28x28x1
14x14x32
7x7x64
3x3x128
1152
10
1152
3x3x128
7x7x64
14x14x32
28x28x1
Conv1
stride=2
Conv2
stride=2
Conv3
stride=2
Flatten
h
FC
Reshape
DeConv3
stride=2
DeConv2
stride=2
DeConv1
stride=2

The Simplest Autoencoder

- Singular Value Decomposition (SVD)
- **Principal Component Analysis (PCA)**
- Karhunen-Loeve transform (KLT)
- Hotelling transform
- Proper Orthogonal Decomposition (POD)
- Eckart-Young Theorem
- Empirical Orthogonal Function (EOF)
- Empirical Eigenfunction Decomposition
- Empirical Component Analysis
- Quasiharmonic Modes
- Spectral Decomposition
- Emperical Modal Analysis

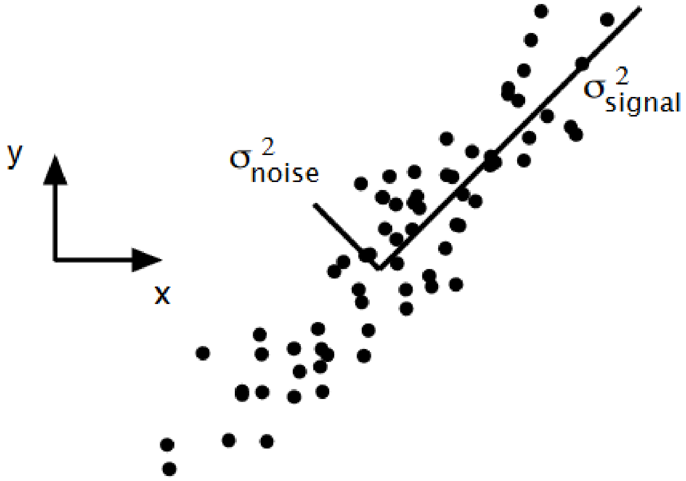
- Some datasets have “bad” coordinates
- We wish to automate finding “good” coordinates



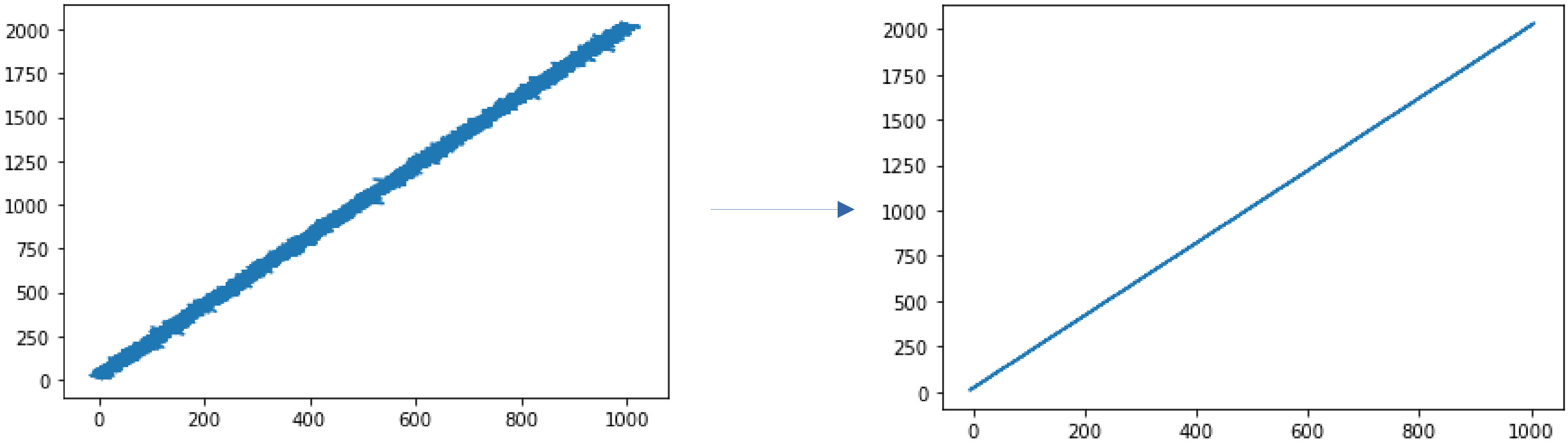
Shlens (2014)
https://arxiv.org/pdf/1404.1100.pdf

Principal Component Analysis

- Question: Are there other coordinates that "better" re-express our dataset
- PCA Finds the best linear change of basis



Shlens (2014)
https://arxiv.org/pdf/1404.1100.pdf


2000
1750
1500
1250
1000
750
500
250
0
0
200
400
600
800
1000
2000
1750
1500
1250
1000
750
500
250
0
0
200
400
600
800
1000

Computation of PCA

(without justification)

- Given a data matrix $\mathbf{X}$
- Let $\mathbf{C_X}$ be its covariance matrix
- The principal components of $\mathbf{X}$ are the eigenvectors of $\mathbf{C_X}$
- The corresponding eigenvalues are the variances along that principal component
- Project $\mathbf{X}$ onto a subset of its principal components

$$\mathbf{X} = \begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ & & \vdots & & \end{bmatrix}$$

$$\mathbf{C_X} = \frac{1}{n-1}(\mathbf{X}-\mu)^T(\mathbf{X}-\mu) = \begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{bmatrix}$$

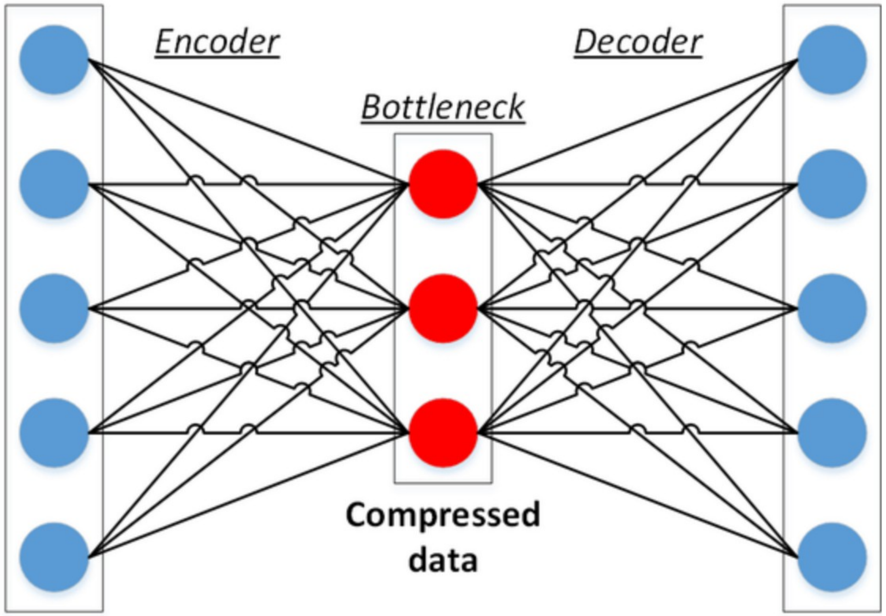
$$= \mathbf{Q\Lambda Q}^T$$

$$\mathbf{Q} = \begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{bmatrix} \qquad \mathbf{P} = \begin{bmatrix} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{bmatrix}$$

$$z = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \end{bmatrix} \qquad \mathbf{P}^T(z-\mu) = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \end{bmatrix} \qquad \mathbf{P}(\mathbf{P}^T(z-\mu)) + \mu = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \end{bmatrix}$$

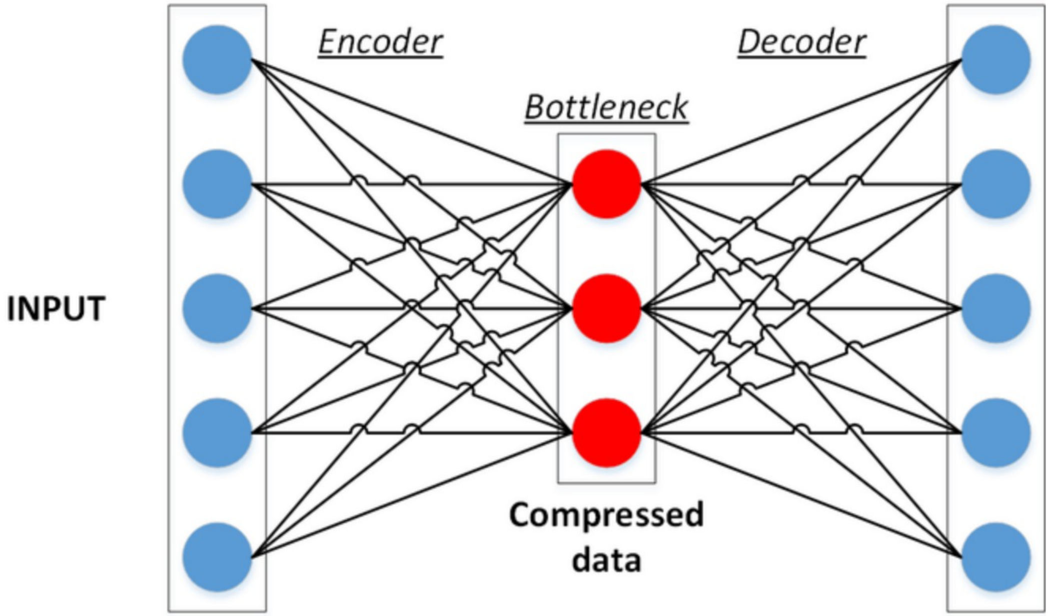
PCA is a linear autoencoder

$$z = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \end{bmatrix} \qquad \mathbf{P}^T(z - \mu) = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \end{bmatrix} \qquad \mathbf{P}(\mathbf{P}^T(z - \mu)) + \mu = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \end{bmatrix}$$

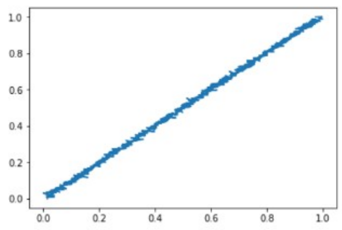
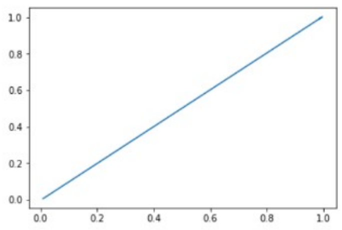
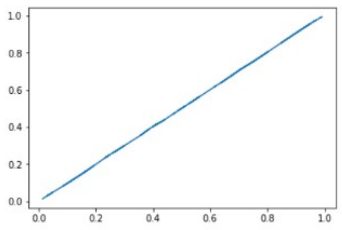
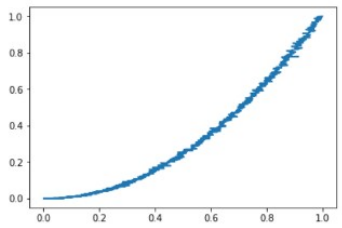
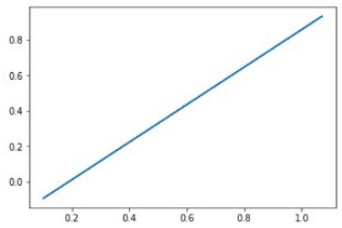
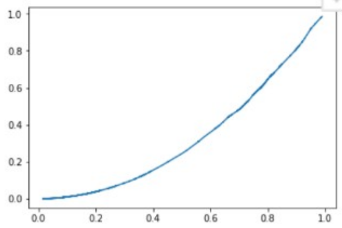
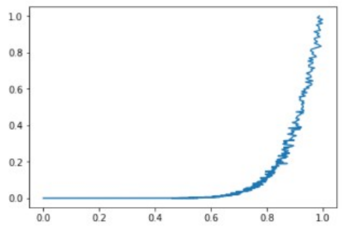
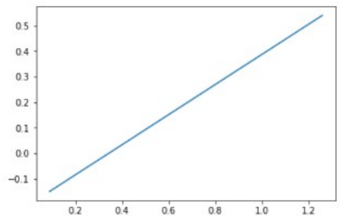
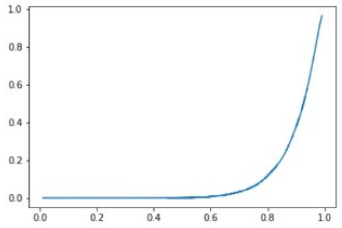


Linear autoencoders “are” PCA

A linear autoencoder with one hidden layer and MSE loss function “is” PCA

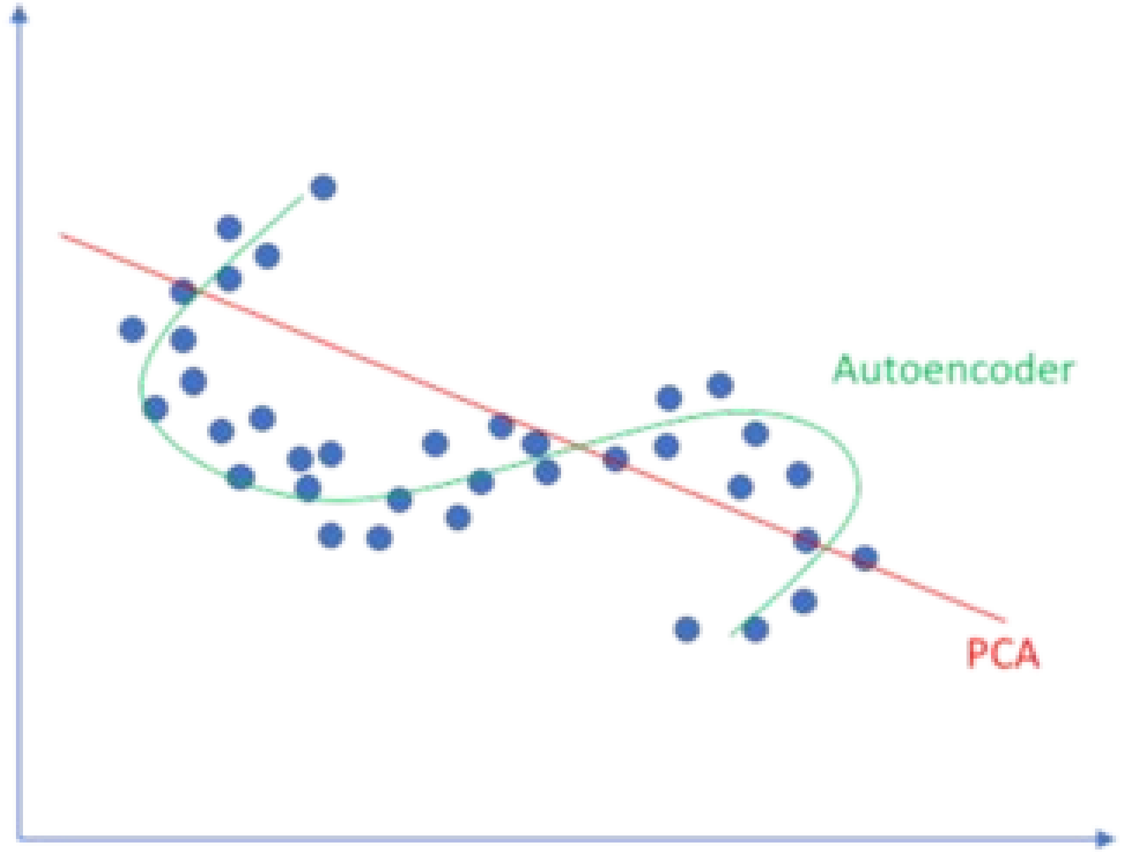


Using non-linear Autoencoders we can relax the linearity assumption

Function	**Feature Space**	**PCA Reconstruction**	**Auto Encoder Reconstruction**
y=mx+c			
$y=mx^2+c$			
$y=mx^8+c$			

https://towardsdatascience.com/autoencoders-vs-pca-when-to-use-which-73de063f5d7

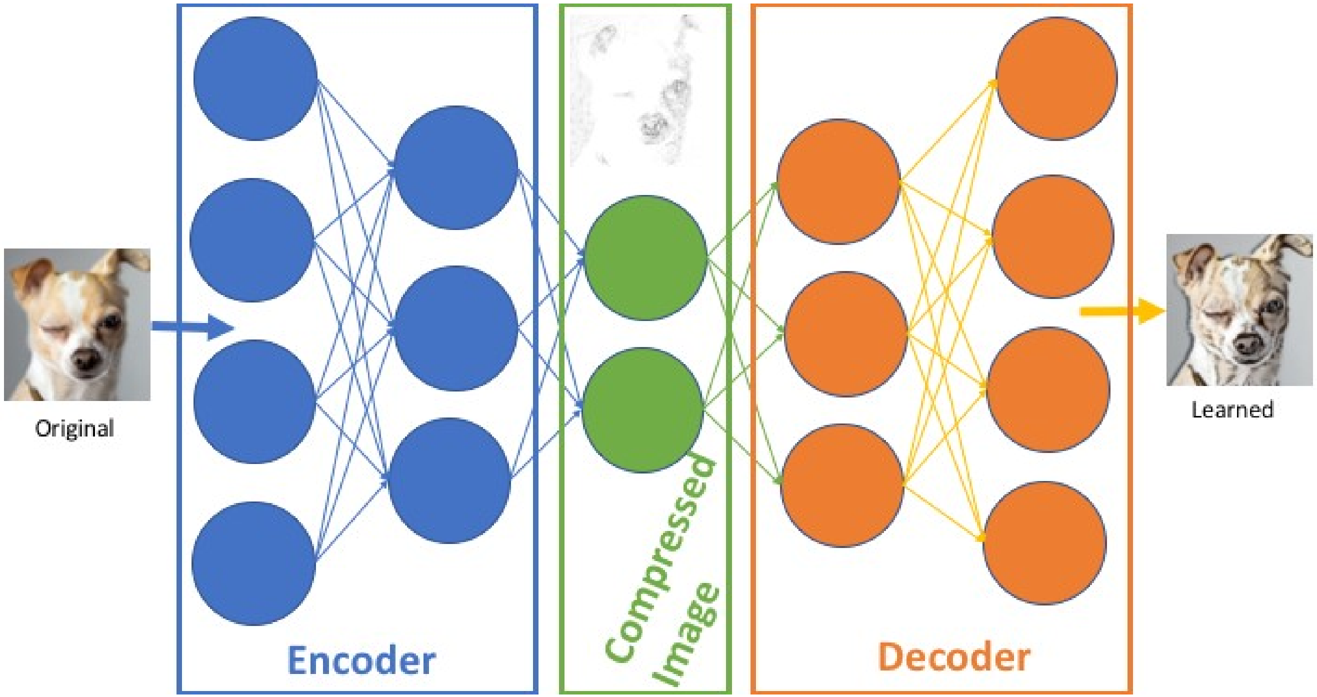
Linear vs nonlinear dimensionality reduction



Applications of Autoencoders

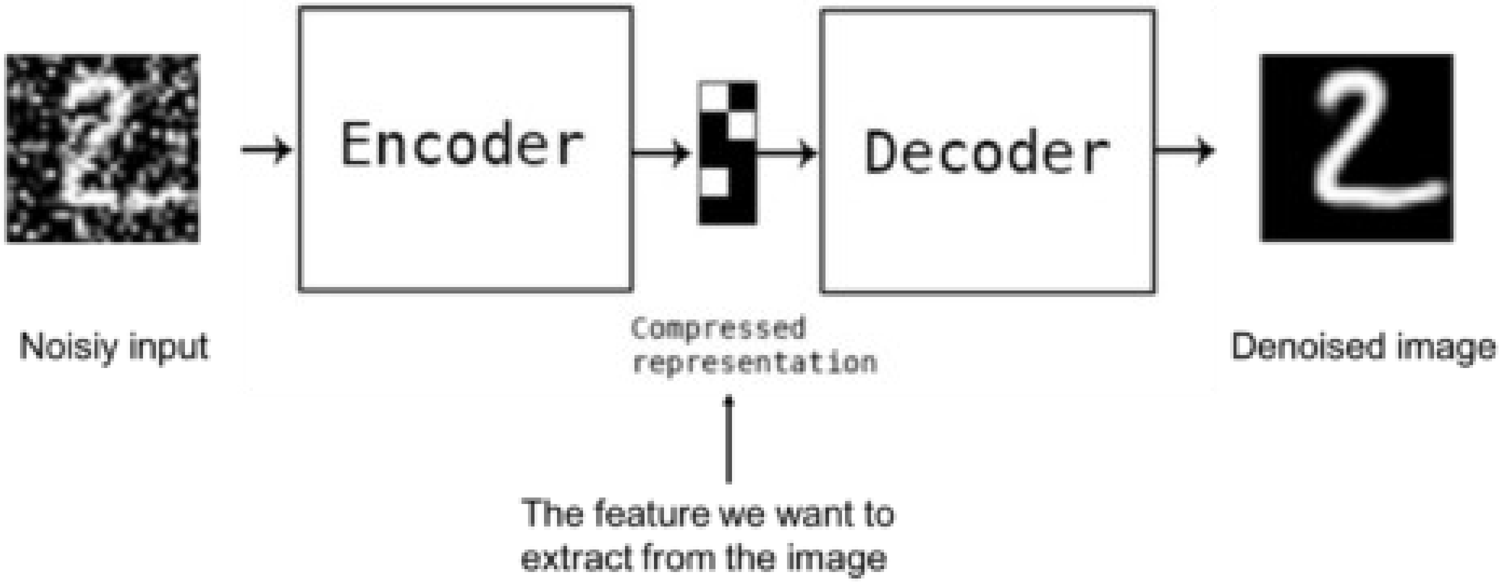
- Image Compression
- Denoising
- Deep fakes
- Generative models (variational autoencoder)
- SINDy
- Recommendation systems
- Anomaly detection
- translation

Image compression



https://arnoldkokoroko.com/projects/imagecompress/

Denoising

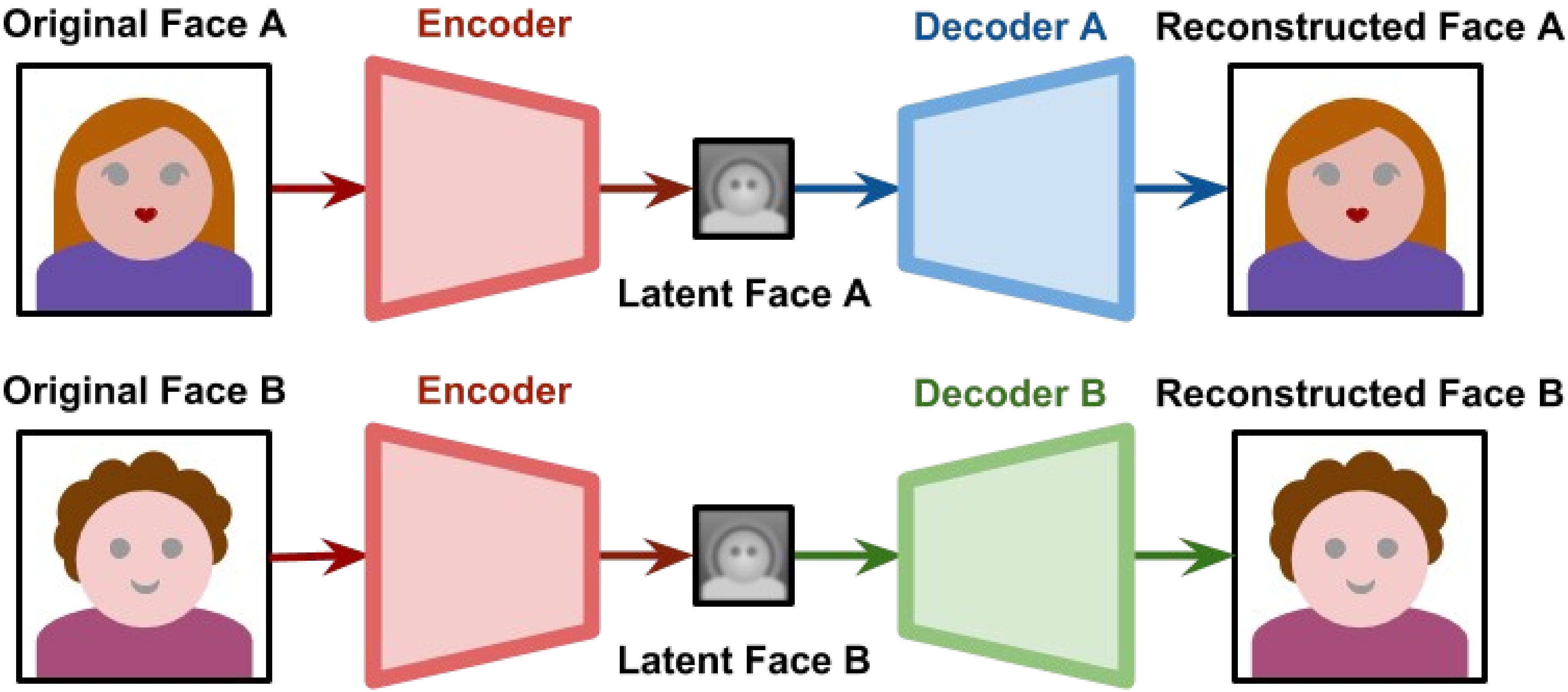


https://medium.com/edureka/autoencoders-tutorial-cfdcebdefe37

DeepFakes

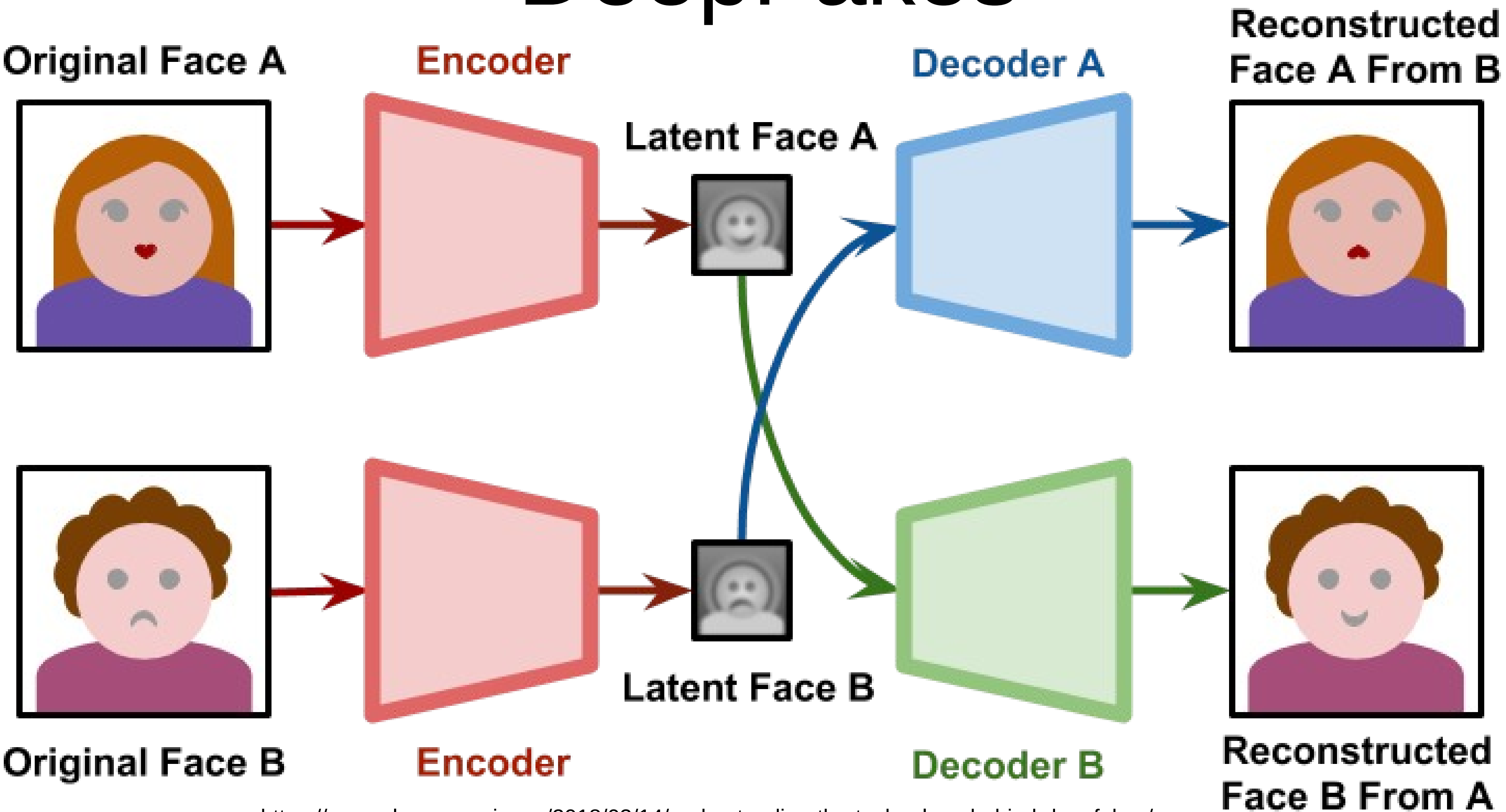
https://www.alanzucconi.com/2018/03/14/understanding-the-technology-behind-deepfakes/

DeepFakes



https://www.alanzucconi.com/2018/03/14/understanding-the-technology-behind-deepfakes/

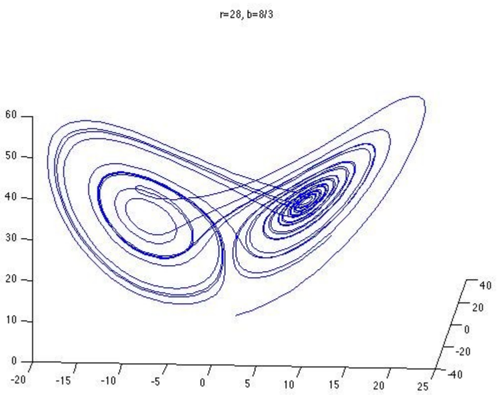
DeepFakes



https://www.alanzucconi.com/2018/03/14/understanding-the-technology-behind-deepfakes/

SINDy

- Sparse Identification of Non-linear Dynamics



$$\frac{\mathrm{d}x}{\mathrm{d}t} = \sigma(y - x),$$

$$\frac{\mathrm{d}y}{\mathrm{d}t} = x(\rho - z) - y,$$

$$\frac{\mathrm{d}z}{\mathrm{d}t} = xy - \beta z.$$

- An autoencoder can help find good coordinates

Brunton, Proctor, and Kutz. 2016, PNAS
https://www.youtube.com/watch?v=NxAn0oglMVw