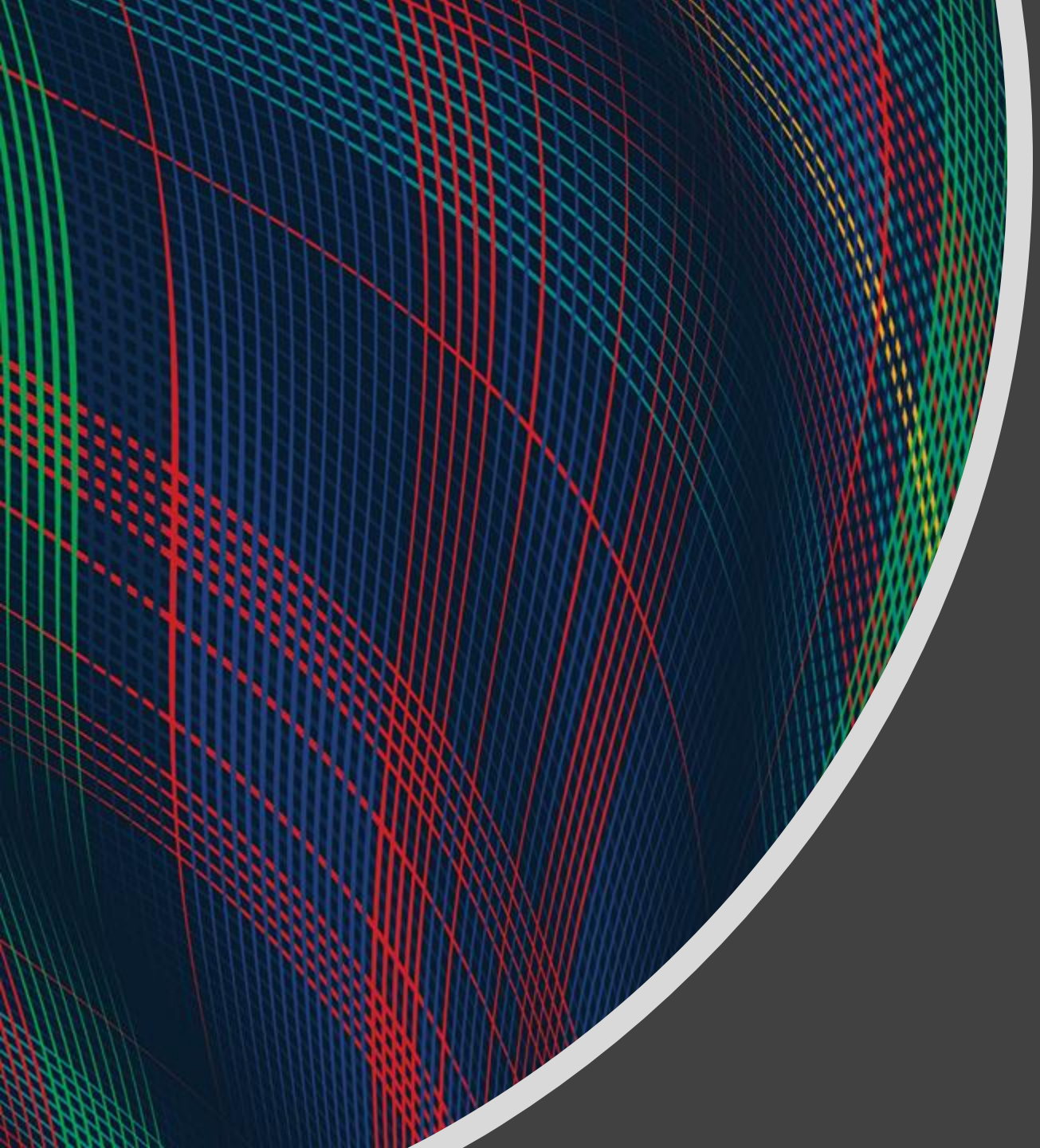


10-315
Introduction to ML

Dimensionality Reduction: PCA, Autoencoders, and Feature Learning

Instructor: Pat Virtue

Learning Paradigms

LM: super $x_{1:t} \rightarrow y_{t+1}$

un/self-super $x_{1:t} \rightarrow x_{t+1}$

Paradigm	Data
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N}$ $\quad \mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
↪ Regression	$y^{(i)} \in \mathbb{R}$
↪ Classification	$y^{(i)} \in \{1, \ldots, K\}$
↪ Binary classification	$y^{(i)} \in \{+1, -1\}$
↪ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ $\quad \mathbf{x} \sim p^*(\cdot)$
Semi-supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N_1} \cup \{\mathbf{x}^{(j)}\}_{j=1}^{N_2}$
Online	$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), (\mathbf{x}^{(3)}, y^{(3)}), \ldots\}$
Active Learning	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ and can query $y^{(i)} = c^*(\cdot)$ at a cost
Imitation Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}), (s^{(2)}, a^{(2)}), \ldots\}$
Reinforcement Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}, r^{(1)}), (s^{(2)}, a^{(2)}, r^{(2)}), \ldots\}$

Self-supervised
Auto-regressive

Slide credit: CMU MLD, Matt Gormley

Outline

Unsupervised Learning

Dimensionality Reduction

Embedded Spaces and Feature Learning

Autoencoders

Principal Component Analysis (PCA)

- Examples: 2D and 3D
- PCA algorithm
- PCA, eigenvectors, and eigenvalues
- PCA objective and optimization

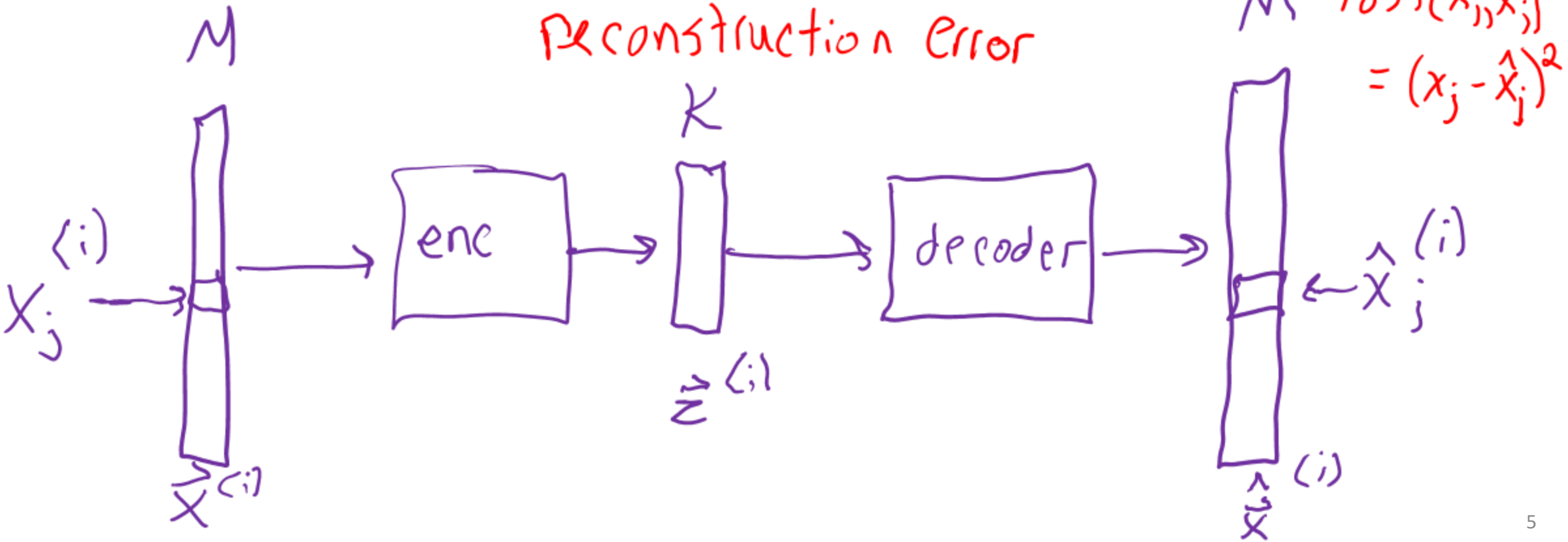
Dimensionality Reduction

Dimensionality Reduction

For each $\mathbf{x}^{(i)} \in \mathbb{R}^M$ find representation $\boldsymbol{z}^{(i)} \in \mathbb{R}^K$ where $K \ll M$

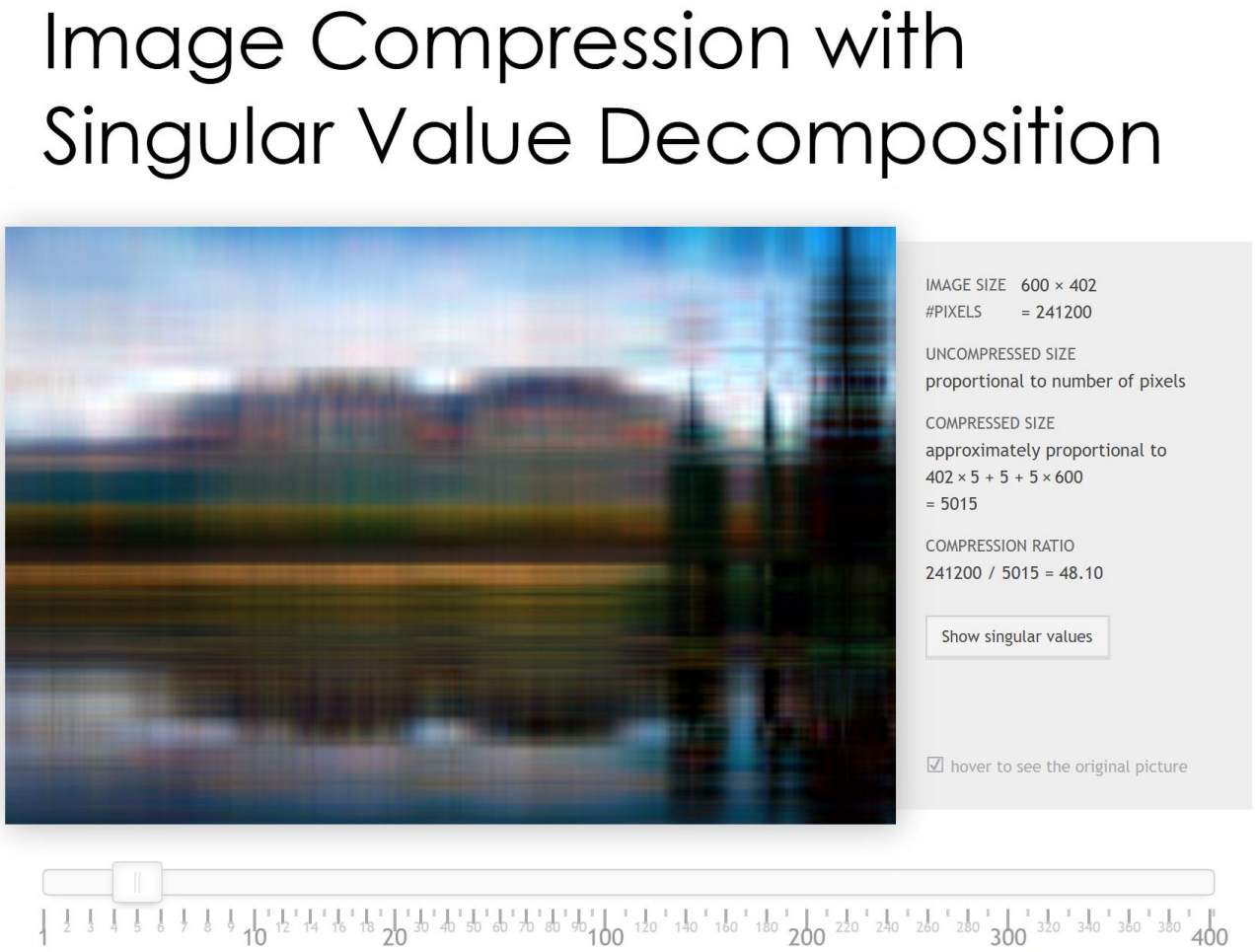
$$\frac{1}{N}\sum_{i=1}^{N} \left\| \vec{x}^{(i)} - \hat{\vec{x}}^{(i)} \right\|_2^2$$

reconstruction error



Dimensionality Reduction

http://timbaumann.info/svd-image-compression-demo/



Dimensionality Reduction

http://timbaumann.info/svd-image-compression-demo/

https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html

Feature Learning

Learning a lower dimensional representation of our data rather than doing feature engineering to represent the data

Also called feature embedding

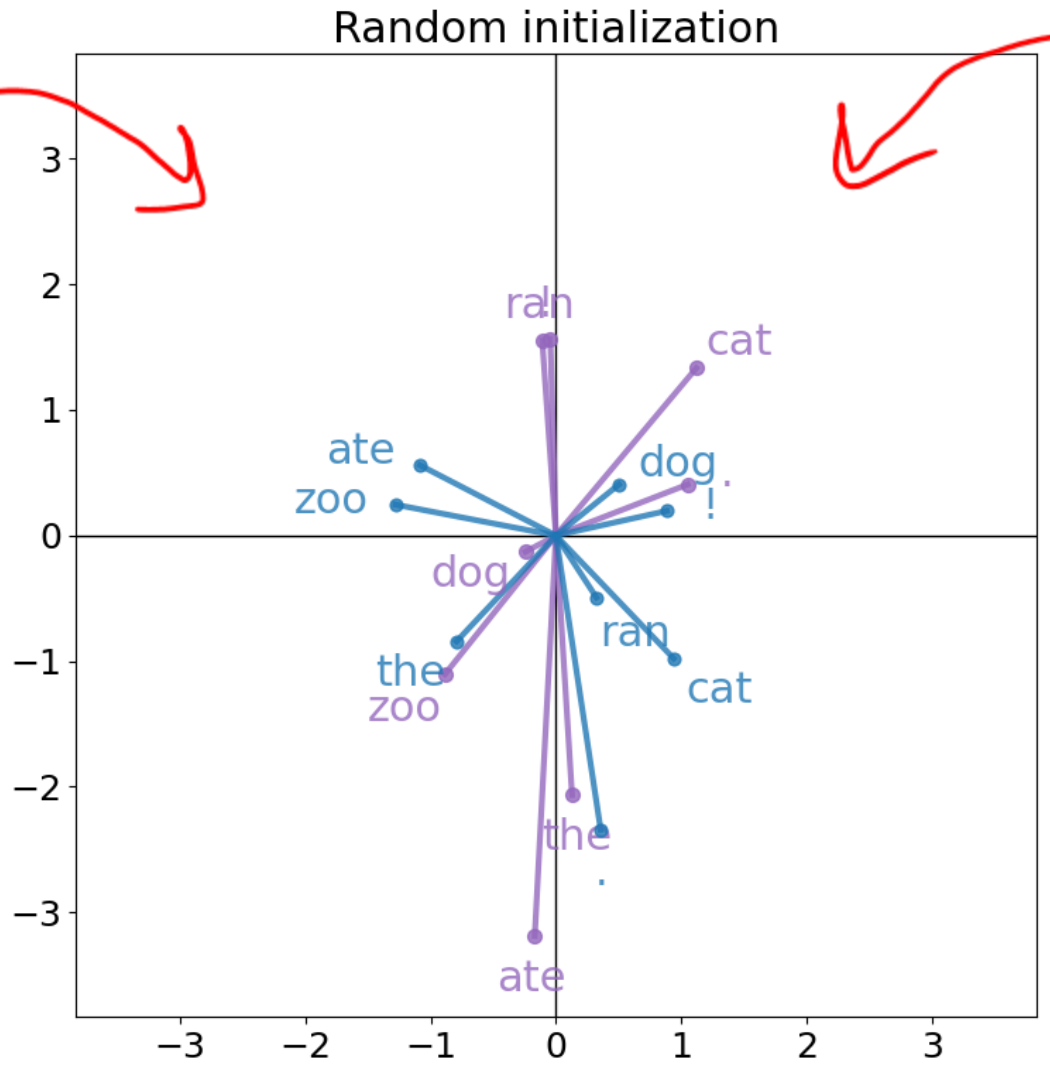
(embedding data in a lower/different dimensional space)

Word Embeddings

Vector representation for each token in vocabulary (initially random)

V: Previous

V:		
!:	0.884,	0.196
.:	0.358,	-2.343
ate:	-1.085,	0.560
cat:	0.939,	-0.978
dog:	0.503,	0.406
ran:	0.323,	-0.493
the:	-0.792,	-0.842
zoo:	-1.280,	0.246

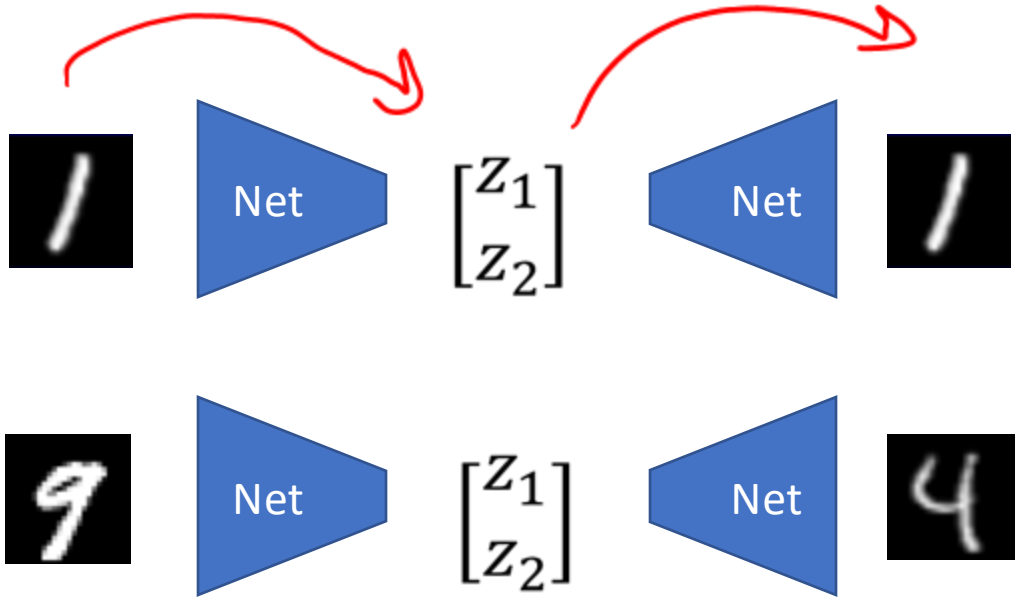


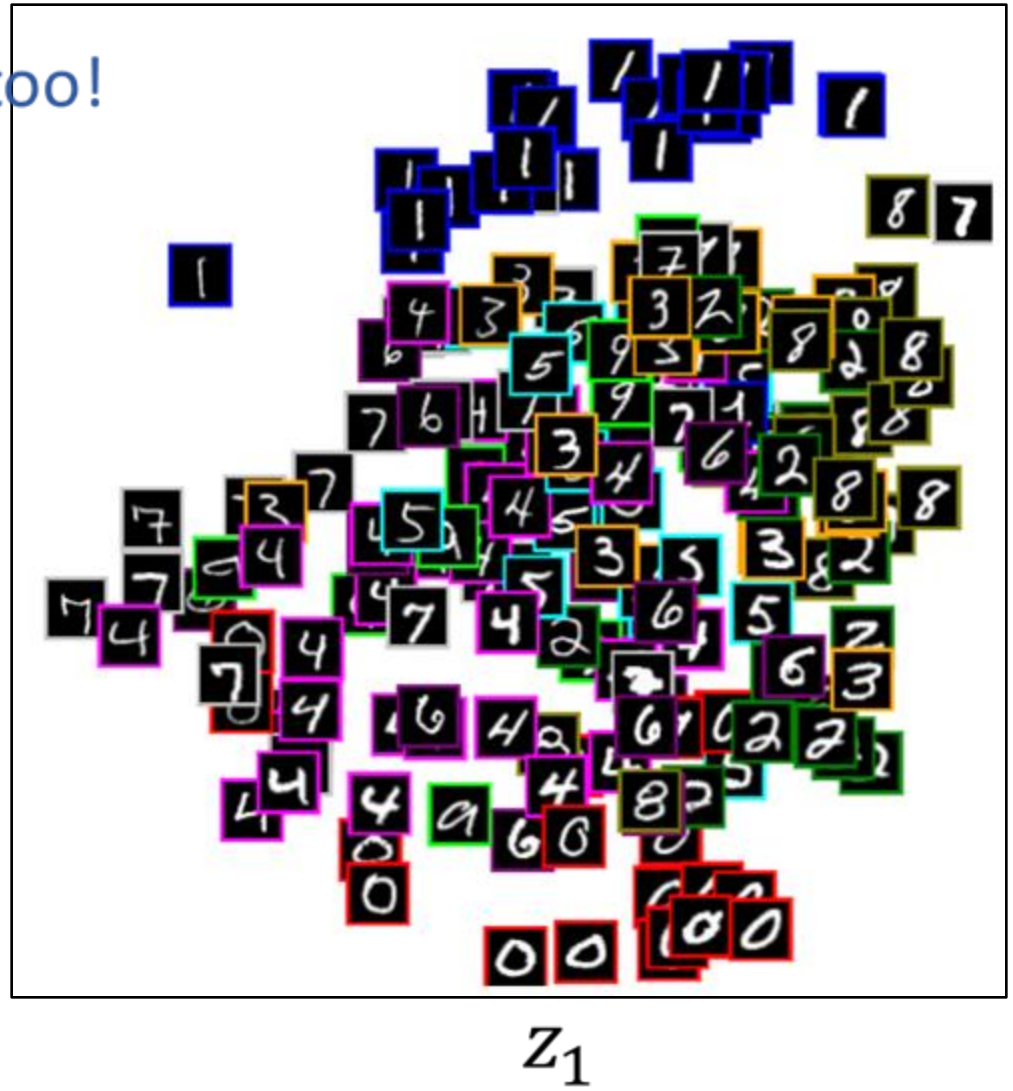
U: Next

U:		
!:	-0.044,	1.568
.:	1.051,	0.406
ate:	-0.169,	-3.190
cat:	1.120,	1.333
dog:	-0.243,	-0.130
ran:	-0.109,	1.556
the:	0.129,	-2.067
zoo:	-0.885,	-1.105

Learning to Organize Data

Neural networks can learn to organization too!

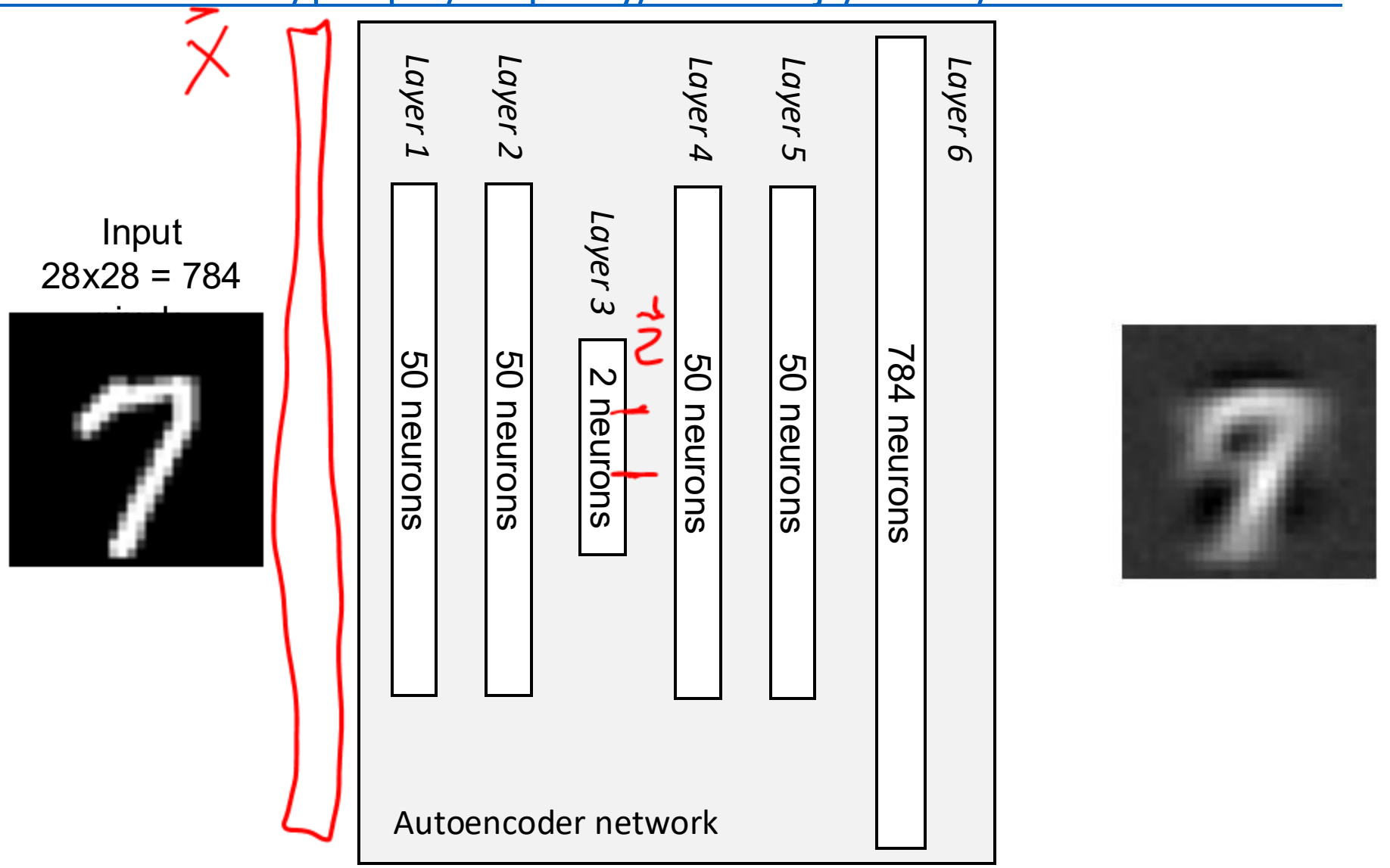




https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html

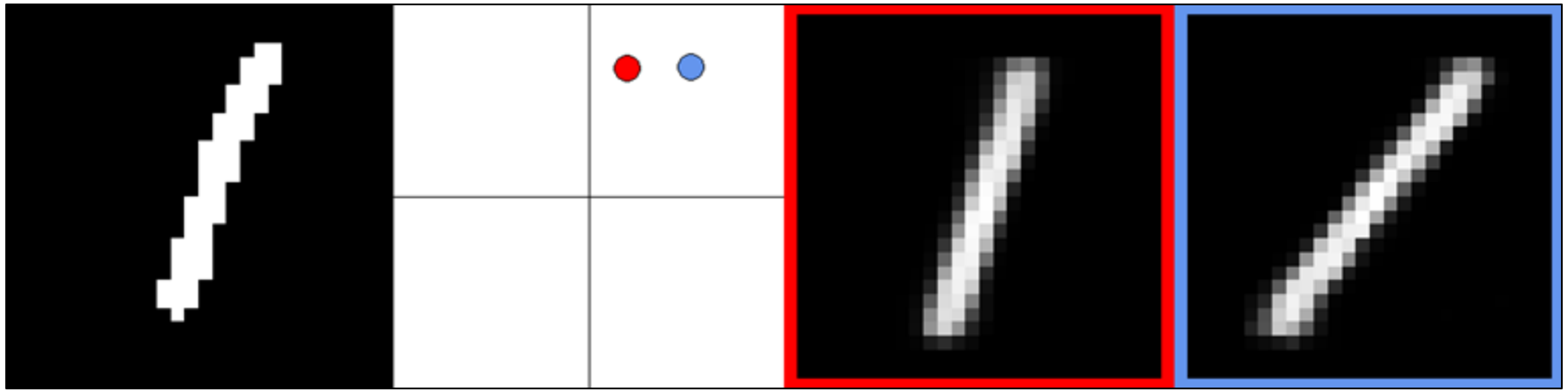
Digit Autoencoder

https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html



Digit Autoencoder

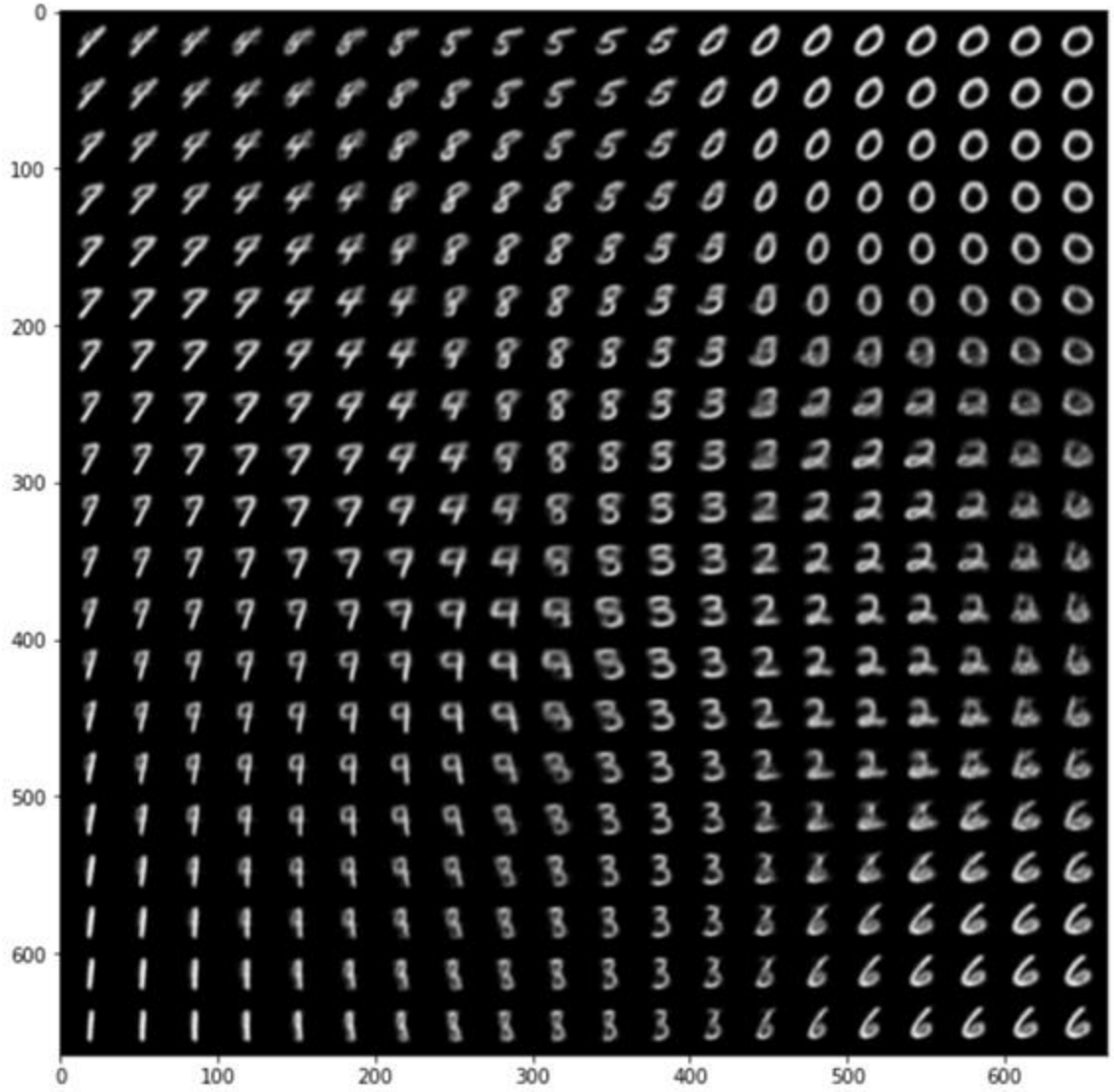
Demo: Using a learned feature space



Variational Autoencoder Demo

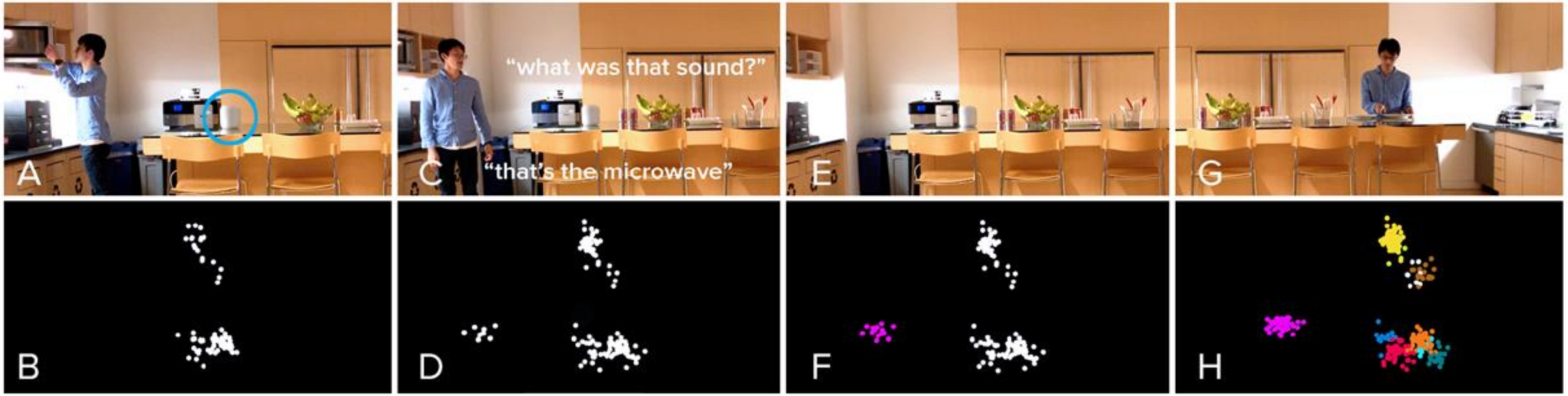
Zhuoyue Lyu, Safinah Ali, and Cynthia Breazeal. EAAI 2022.

https://colab.research.google.com/gist/ZhuoyueLyu/5046225a9ae3675cf633c1df5f63be06/digits-interpolation-notebook-eaai.ipynb



Feature Learning

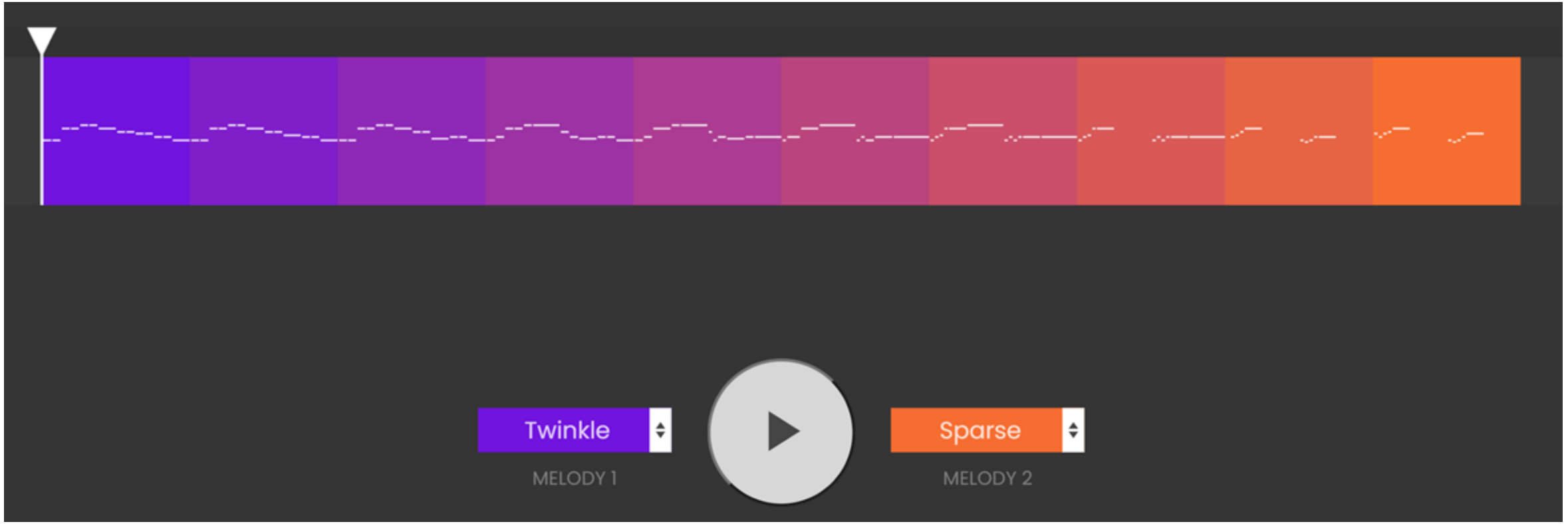
Listen Learner



https://chrisharrison.net/index.php/Research/ListenLearner

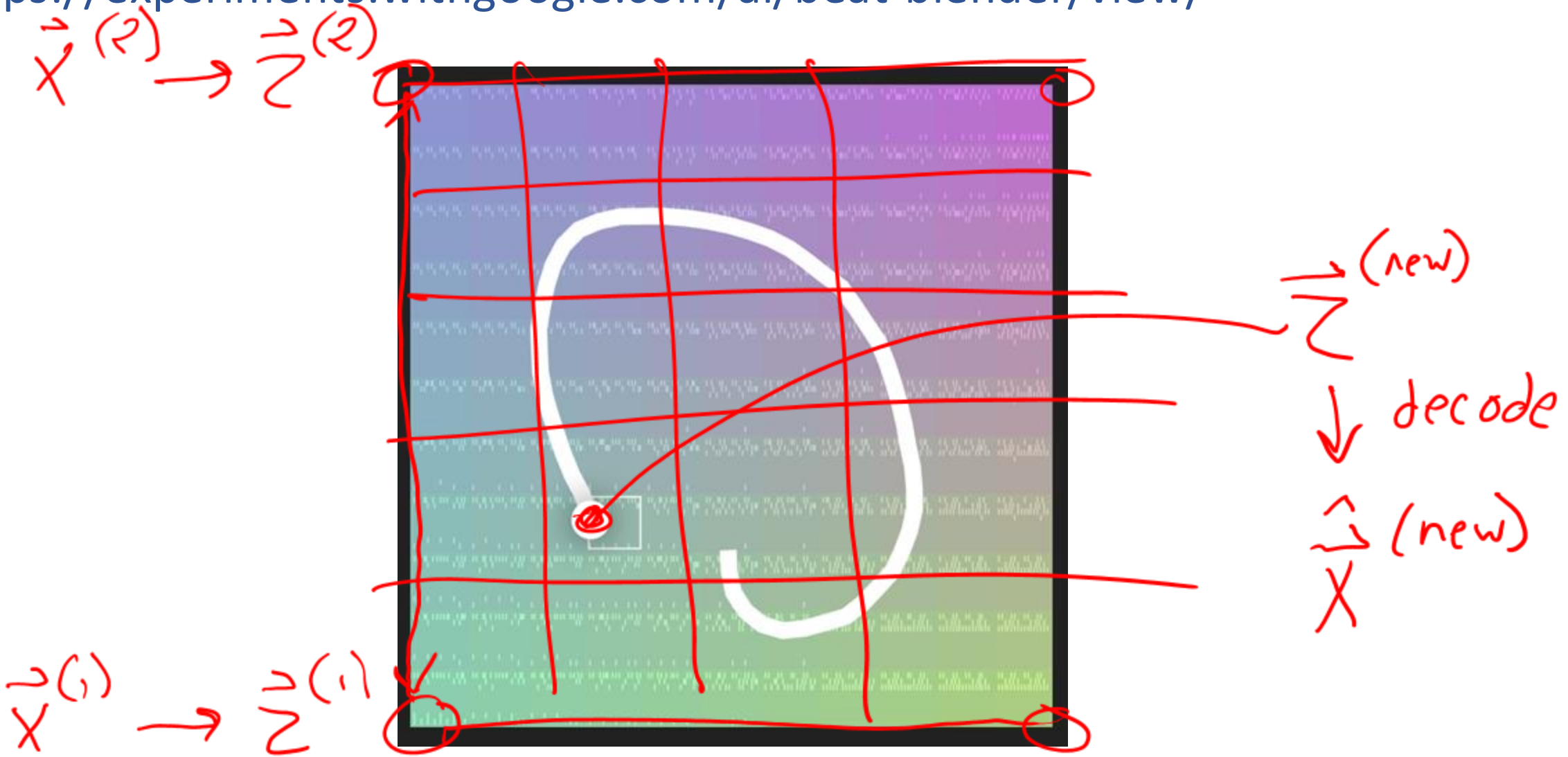
Exploring Feature Space

https://experiments.withgoogle.com/ai/melody-mixer/view/



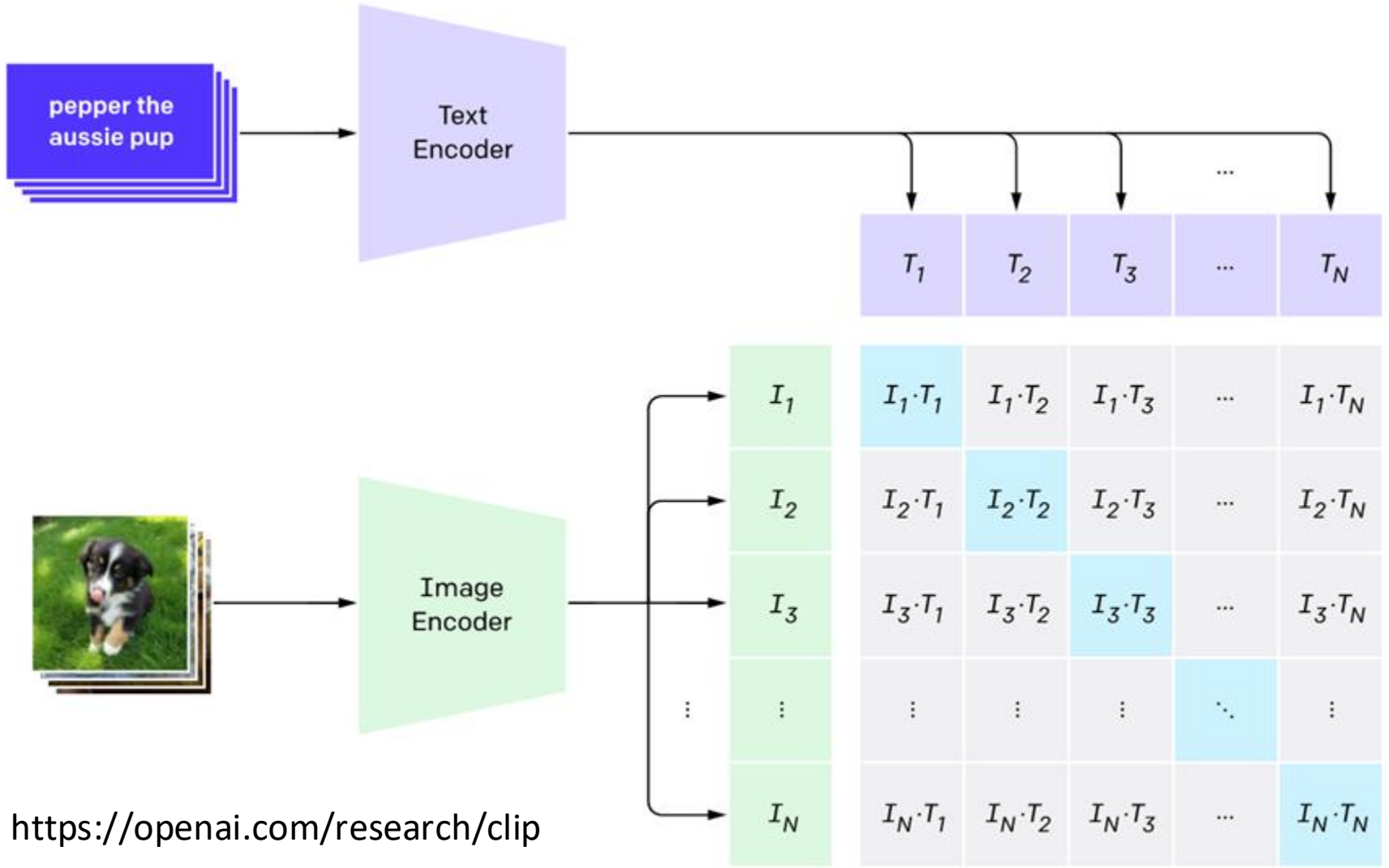
Exploring Feature Space

https://experiments.withgoogle.com/ai/beat-blender/view/



Feature Learning

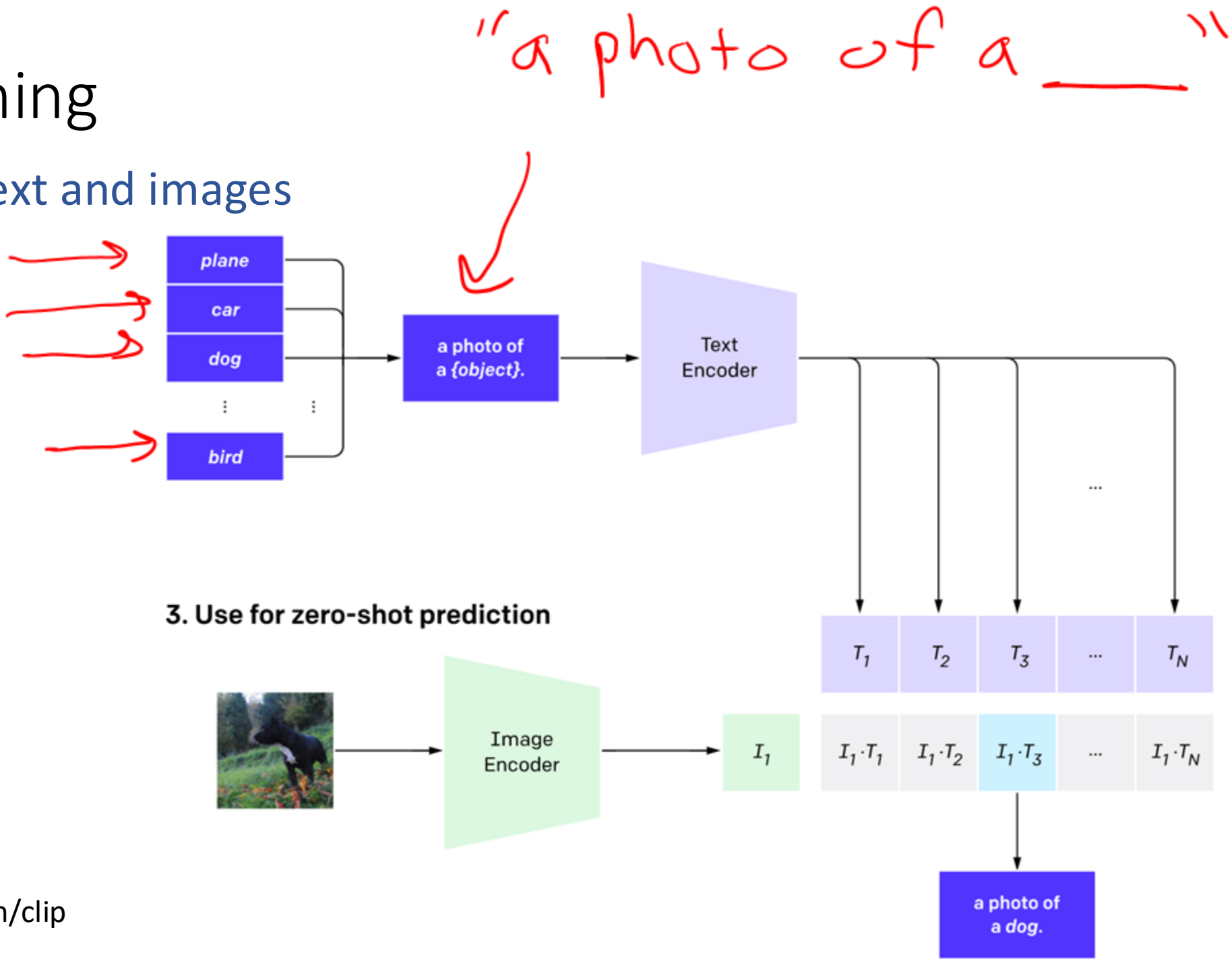
CLIP: Connecting text and images



https://openai.com/research/clip

Feature Learning

CLIP: Connecting text and images



https://openai.com/research/clip

Outline

Unsupervised Learning

Dimensionality Reduction

Embedded Spaces and Feature Learning

Autoencoders

Principal Component Analysis (PCA)

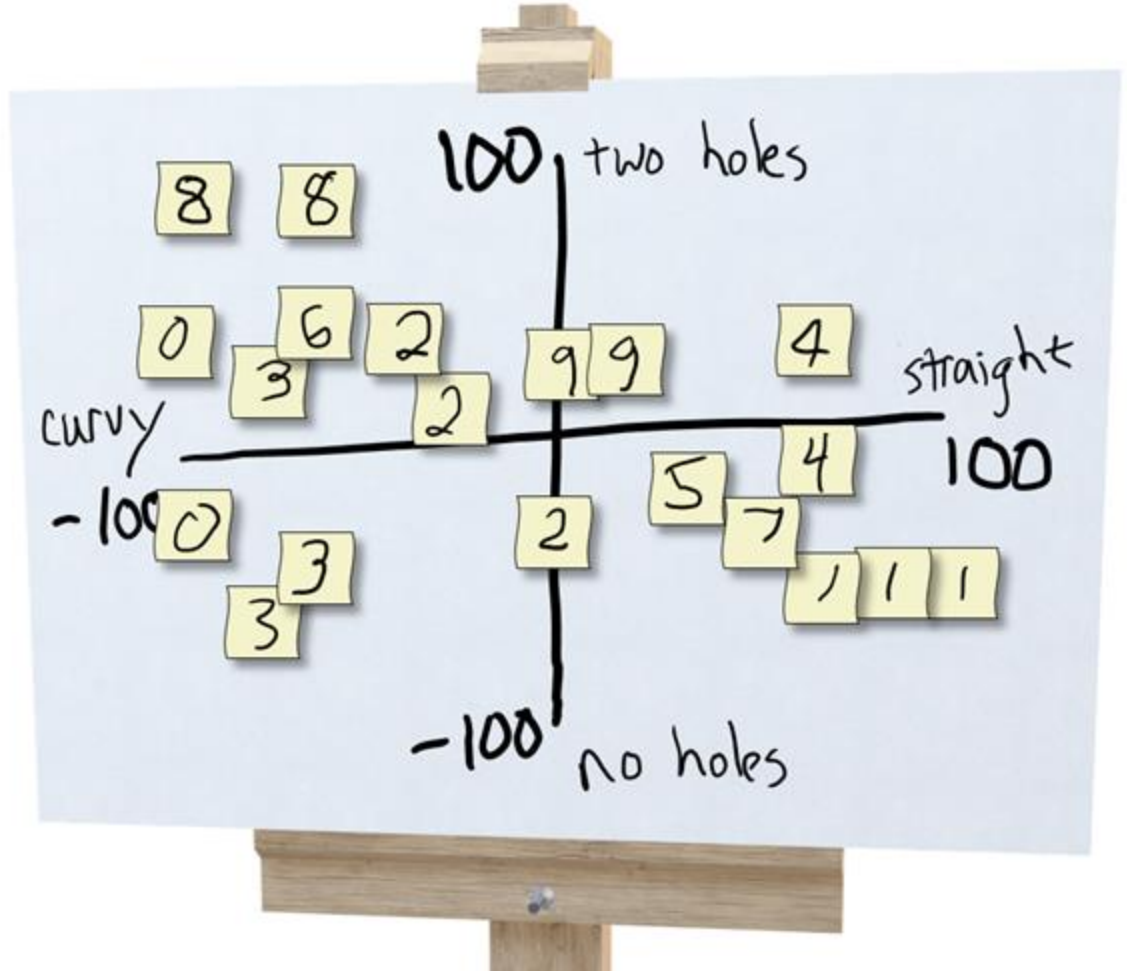
- Examples: 2D and 3D
- PCA algorithm
- PCA, eigenvectors, and eigenvalues
- PCA objective and optimization

Autoencoders

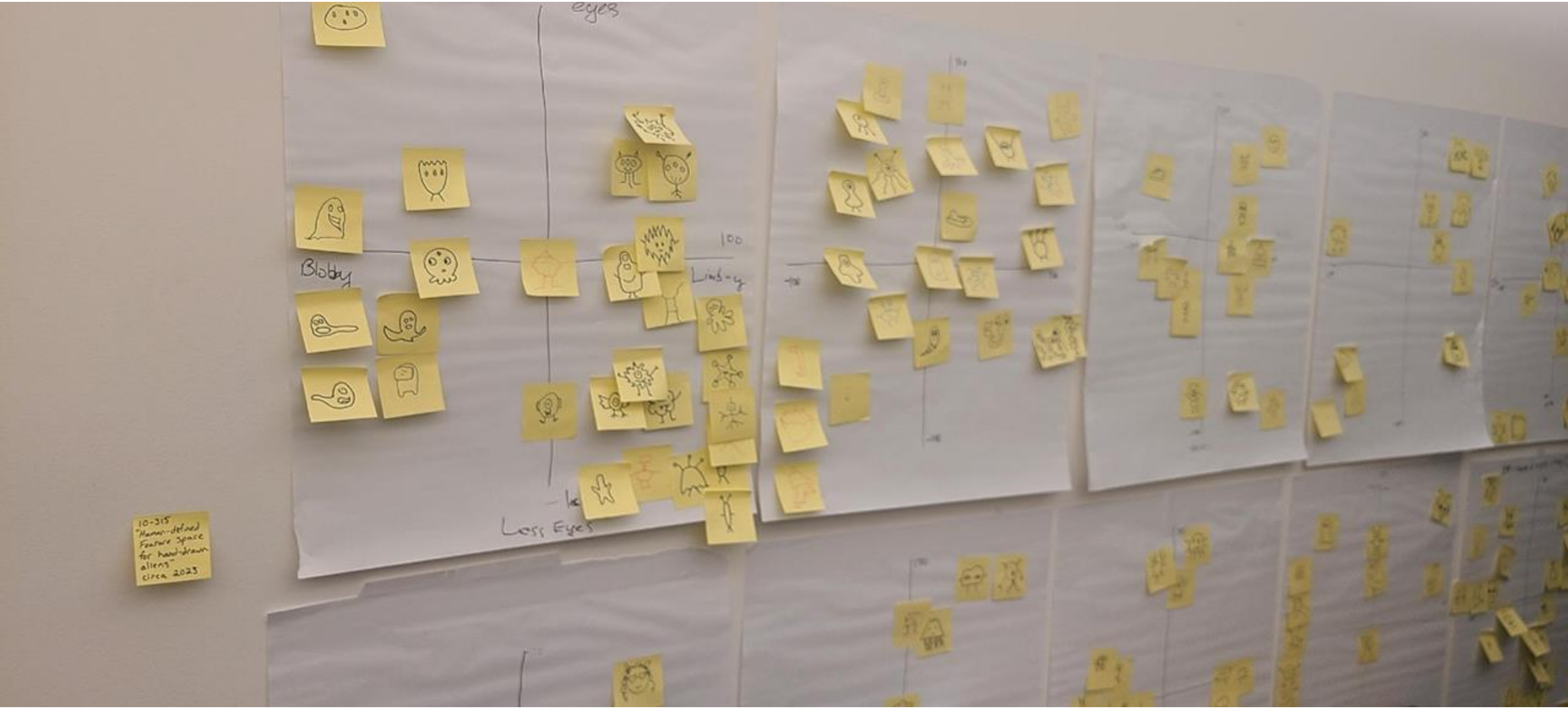
Exercise: Human-defined Feature Space

Step 4: Creation!

1. Select three students: A,B,C
2. Student A draws a new digit and h
3. Student B thinks about where to p
   coordinate, (x, y)
4. Student C looks at the coordinate a
   from A) and draws a new digit

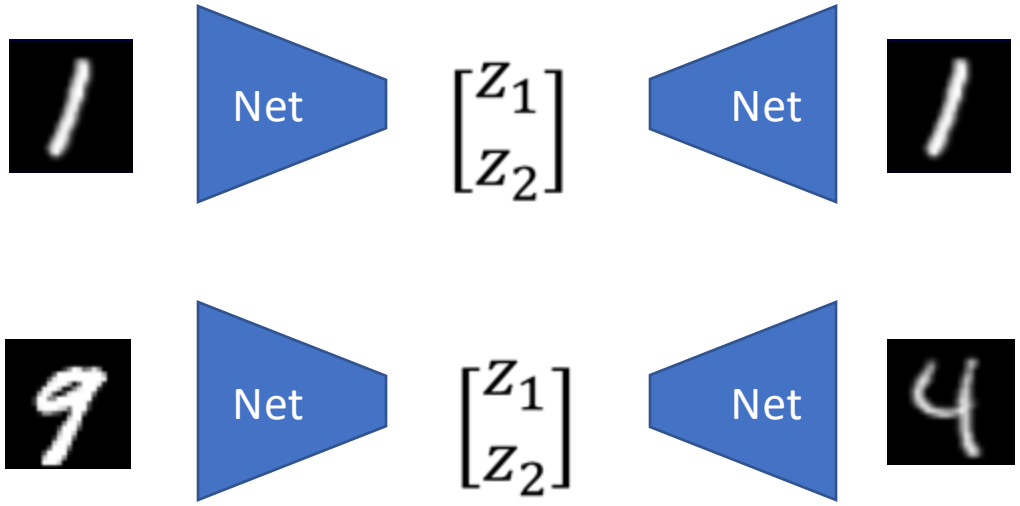


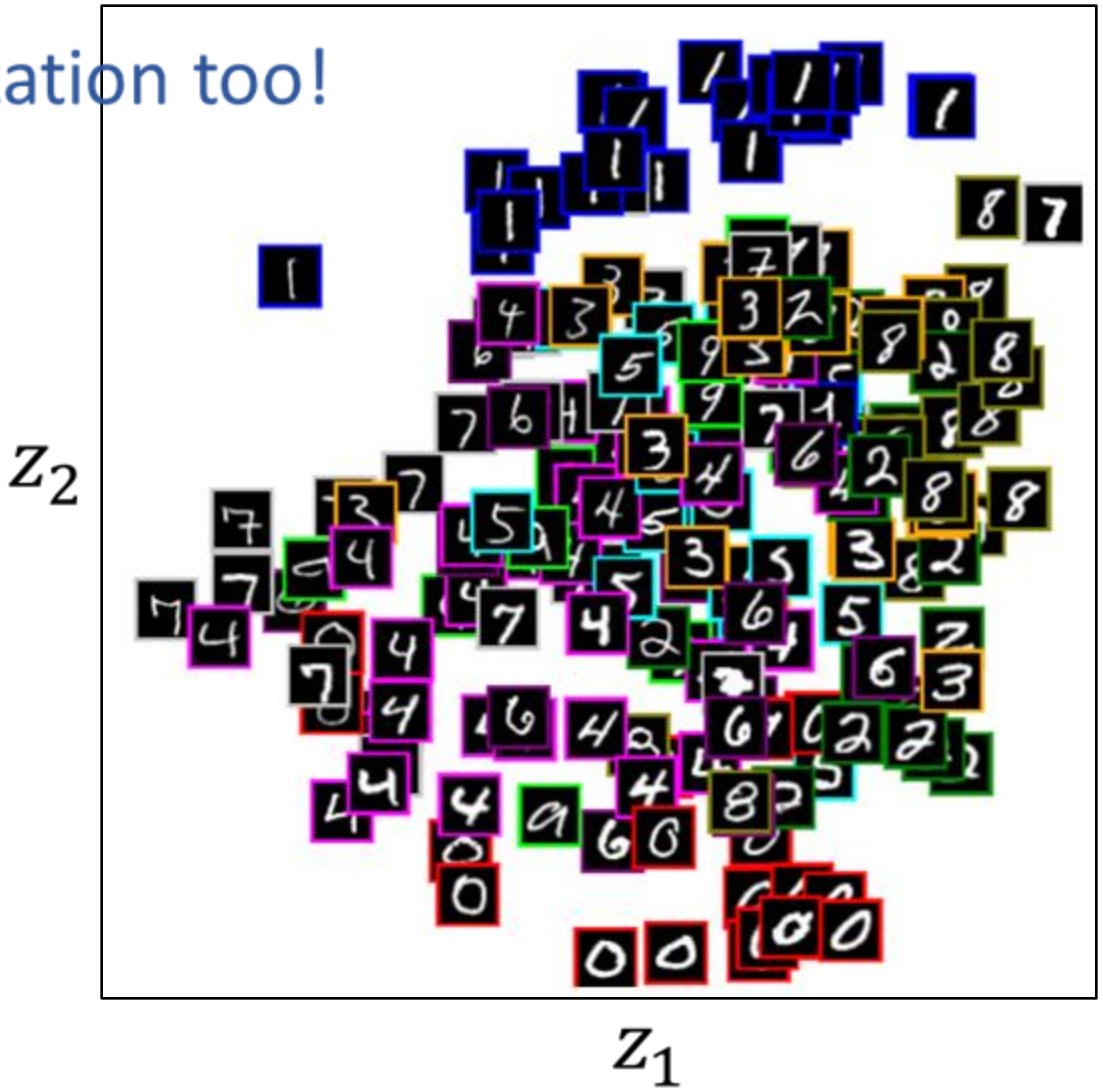
Exercise: Human-defined Feature Space



Learning to Organize Data

Neural networks can learn to organization too!





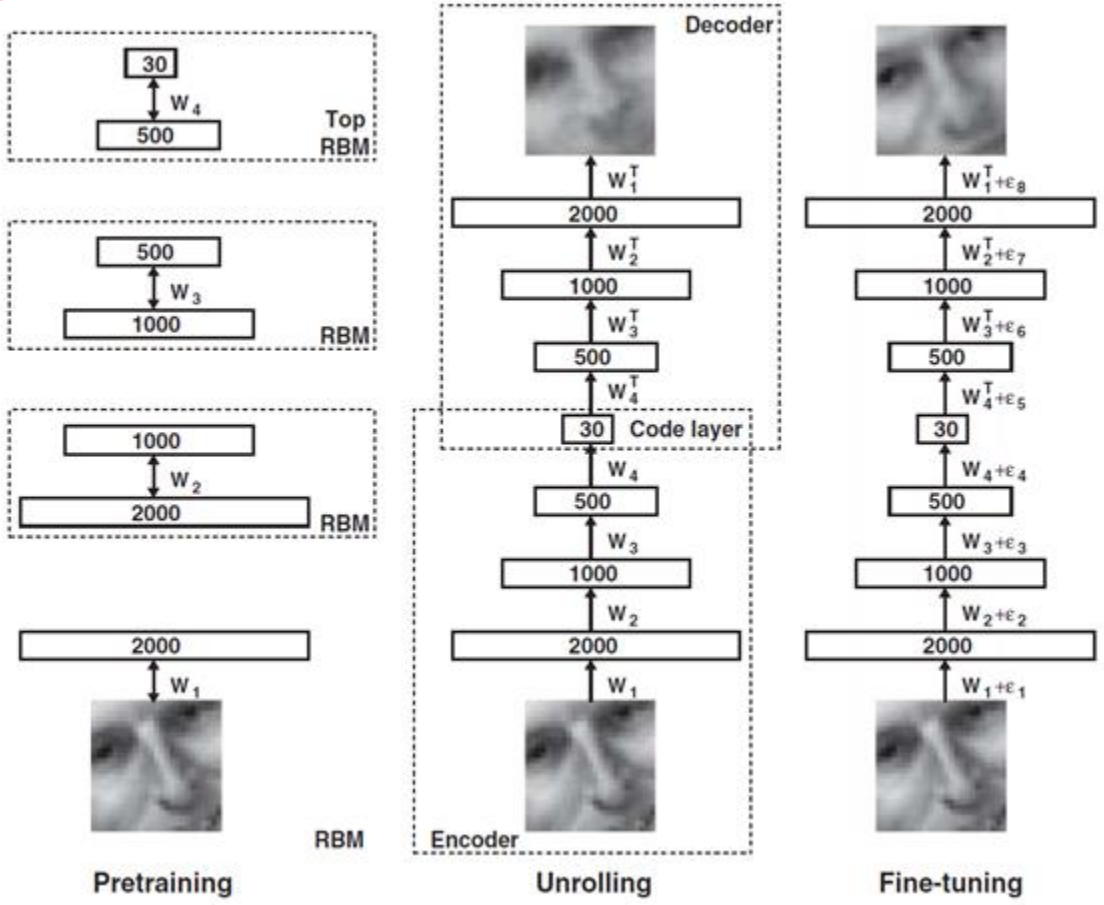
https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html

Dimensionality Reduction with Deep Learning

Hinton, Geoffrey E., and Ruslan R. Salakhutdinov.

"Reducing the dimensionality of data with neural networks."

Science 313.5786 (2006): 504-507.



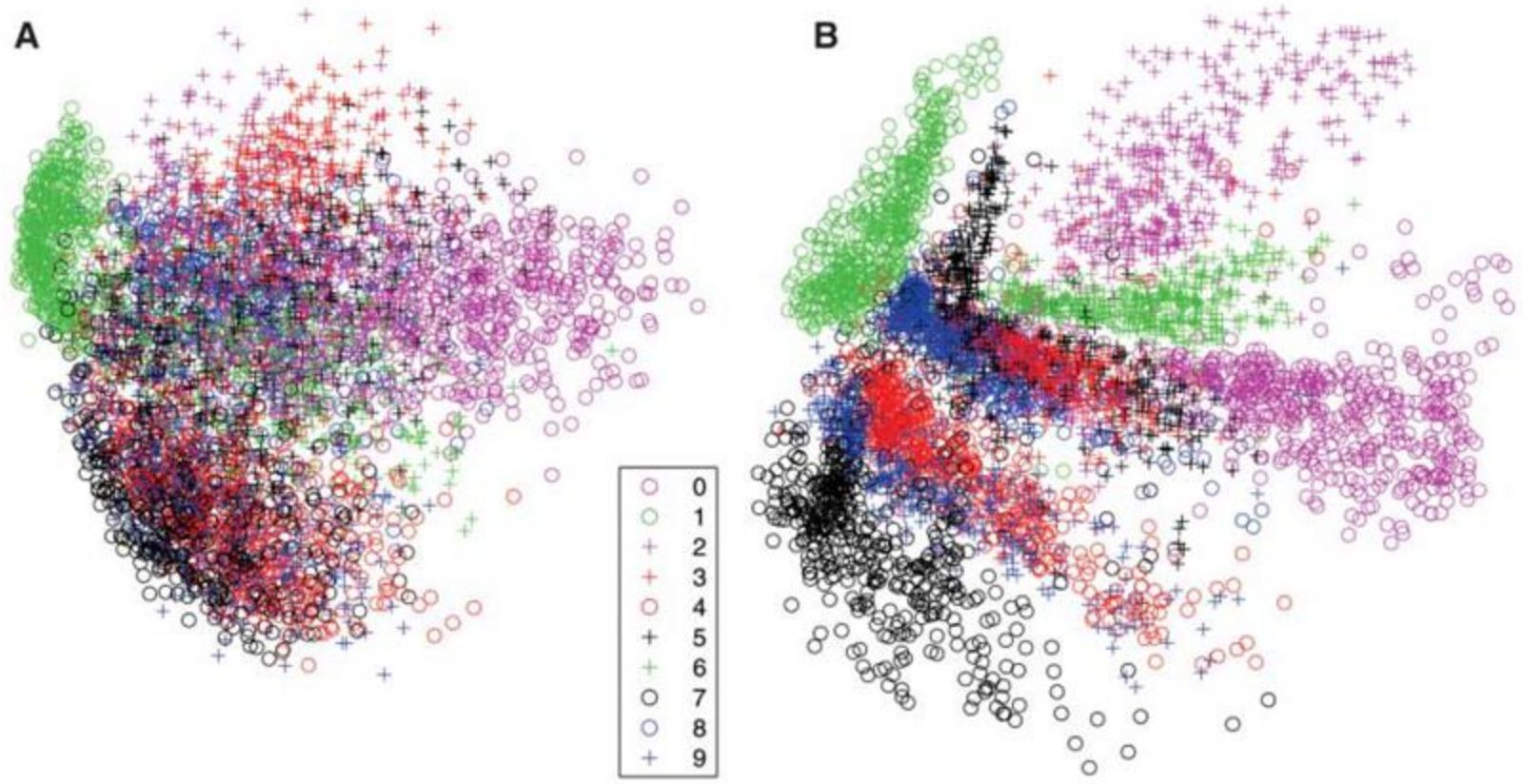
Dimensionality Reduction with Deep Learning

Hinton, Geoffrey E., and Ruslan R. Salakhutdinov.

"Reducing the dimensionality of data with neural networks."

Science 313.5786 (2006): 504-507.

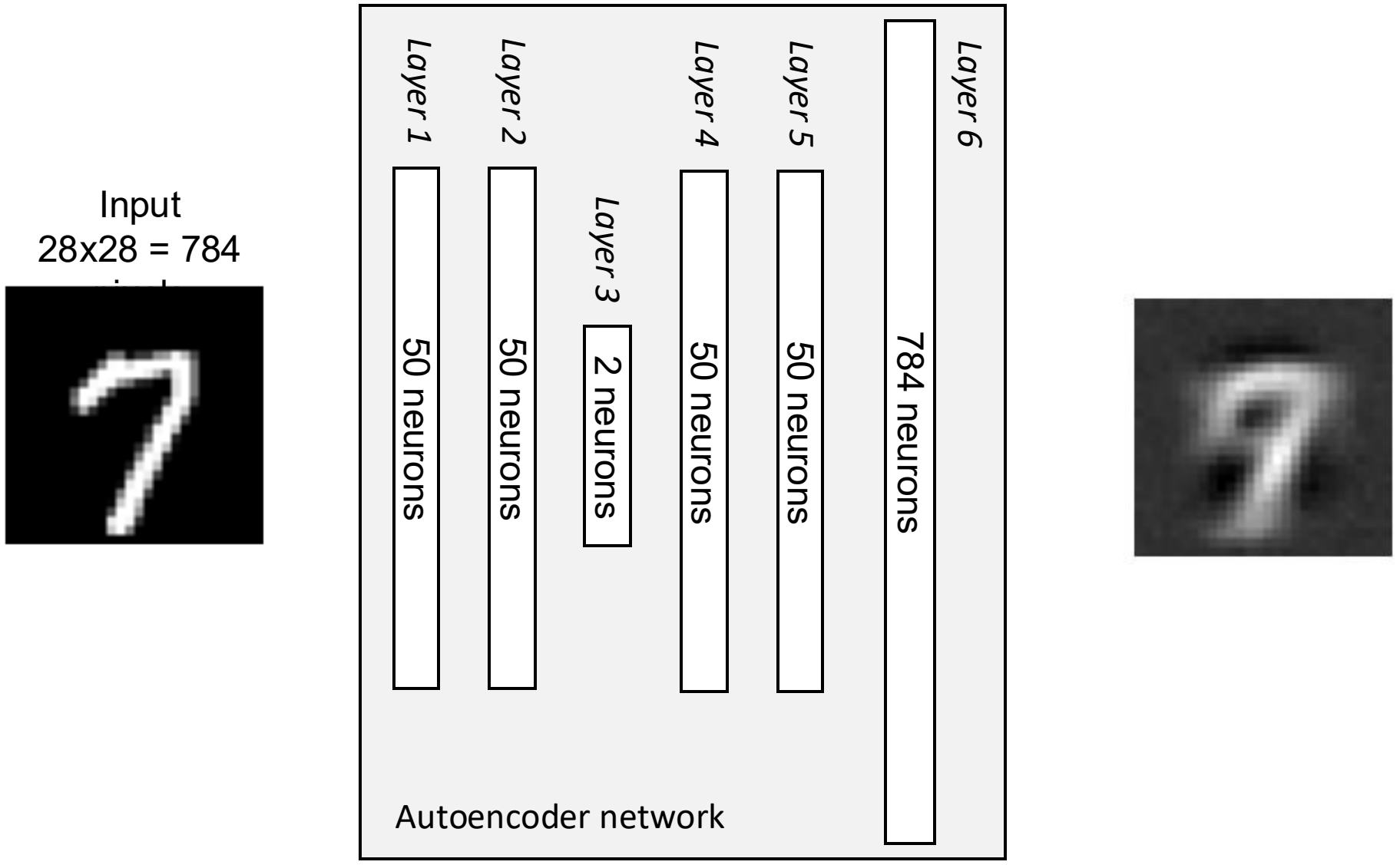
PCA



Neural Network

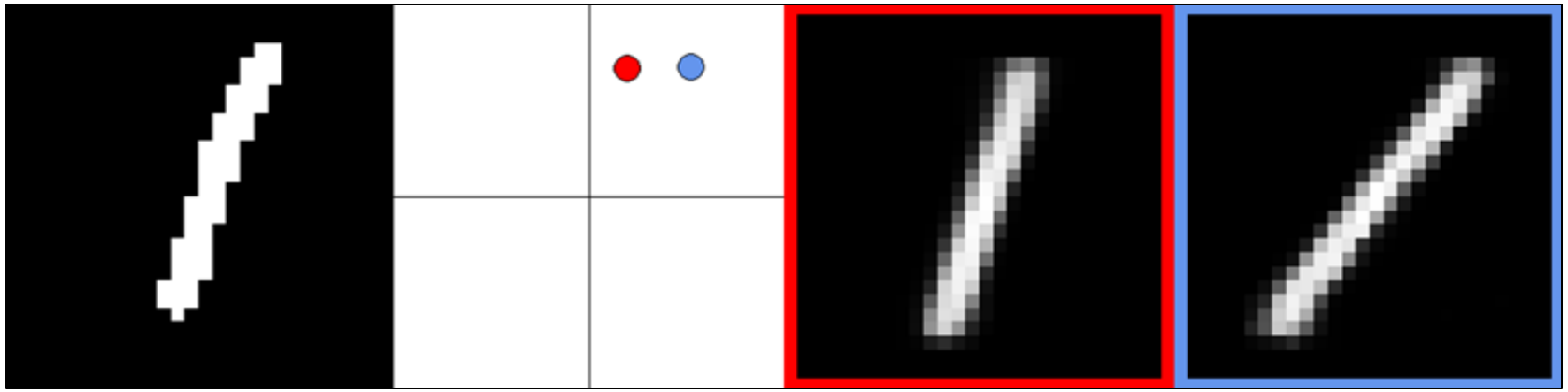
Digit Autoencoder

https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html



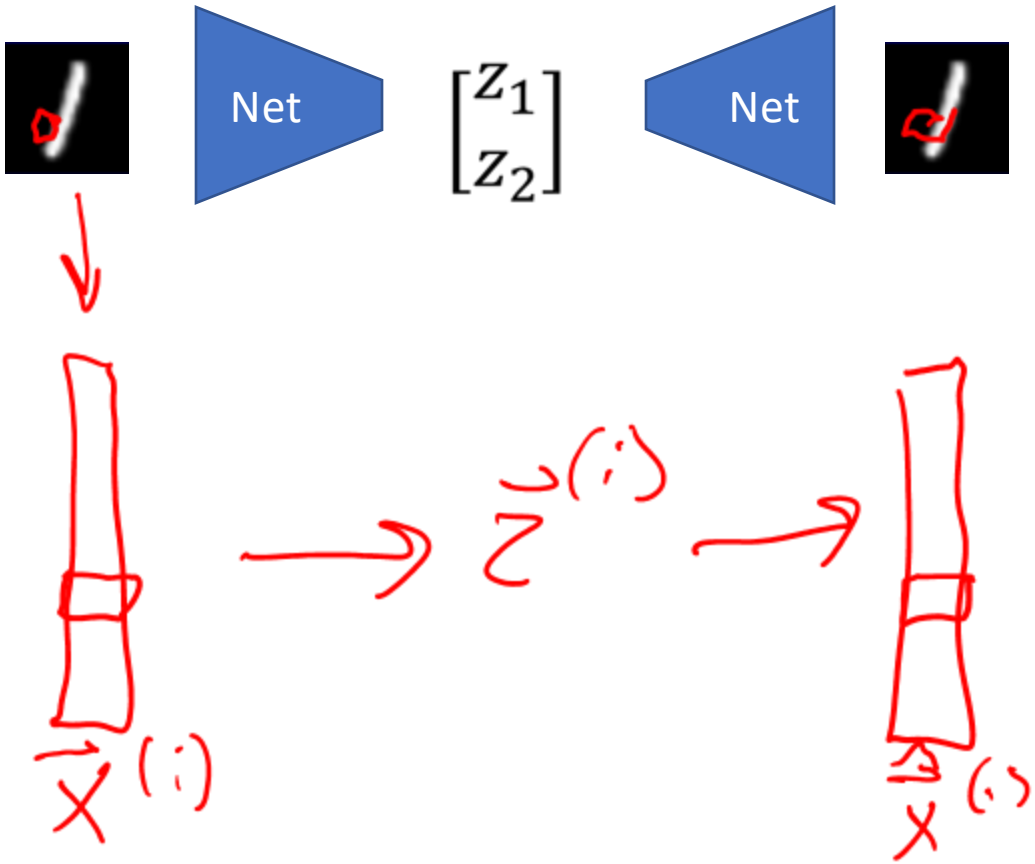
Digit Autoencoder

Demo: Using a learned feature space



Autoencoder objective

Minimize reconstruction error



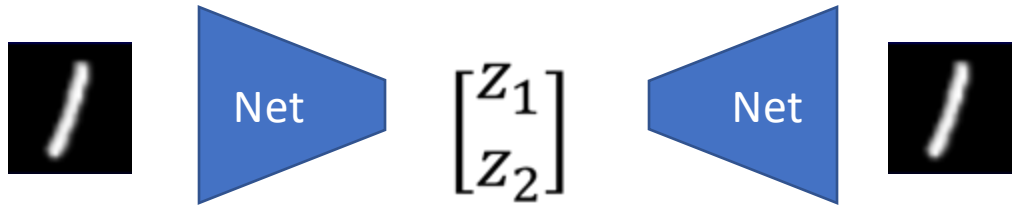
$$\|\vec{x}^{(i)} - \hat{\vec{x}}^{(i)}\|_2^2$$

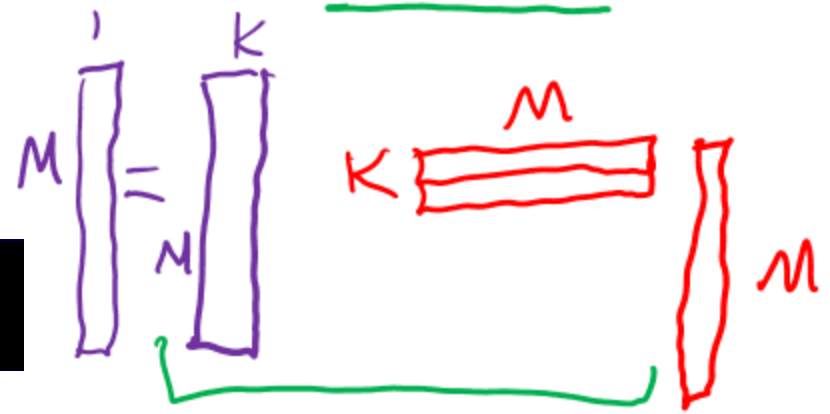
Autoencoder objective

PCA → $\hat{x} = W_2 W_1 x$

$(M \times 1) = (M \times K)(K \times M)(M \times 1)$

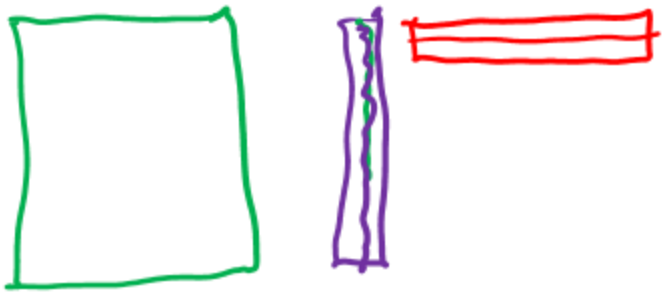
What if networks are just one linear layer?

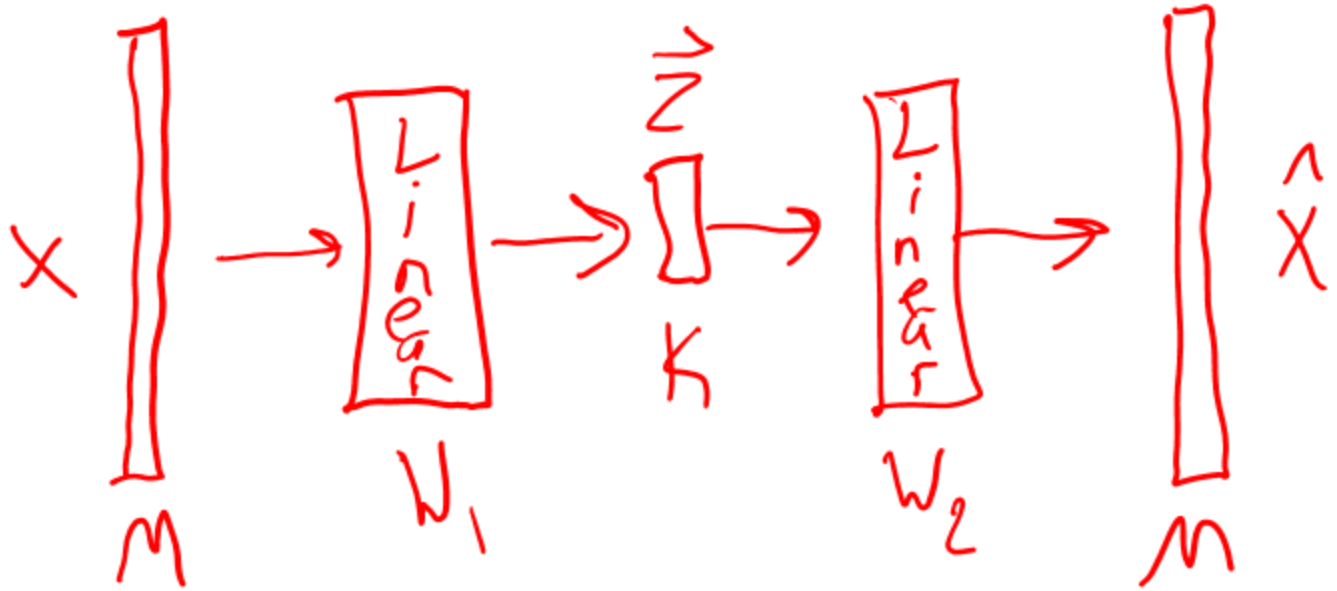




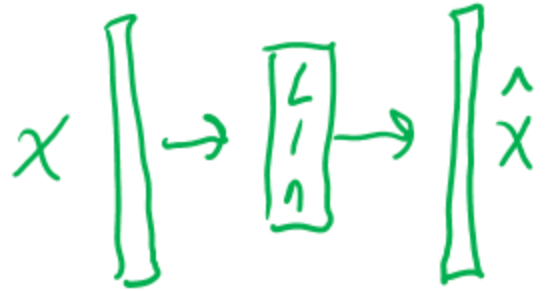
same?

$W_3 = W_2 W_1$





$\hat{x} = W_3 x$



Outline

Unsupervised Learning

Dimensionality Reduction

Embedded Spaces and Feature Learning

Autoencoders

Principal Component Analysis (PCA)

- Examples: 2D and 3D
- PCA algorithm
- PCA, eigenvectors, and eigenvalues
- PCA objective and optimization

Principal Component Analysis (PCA)

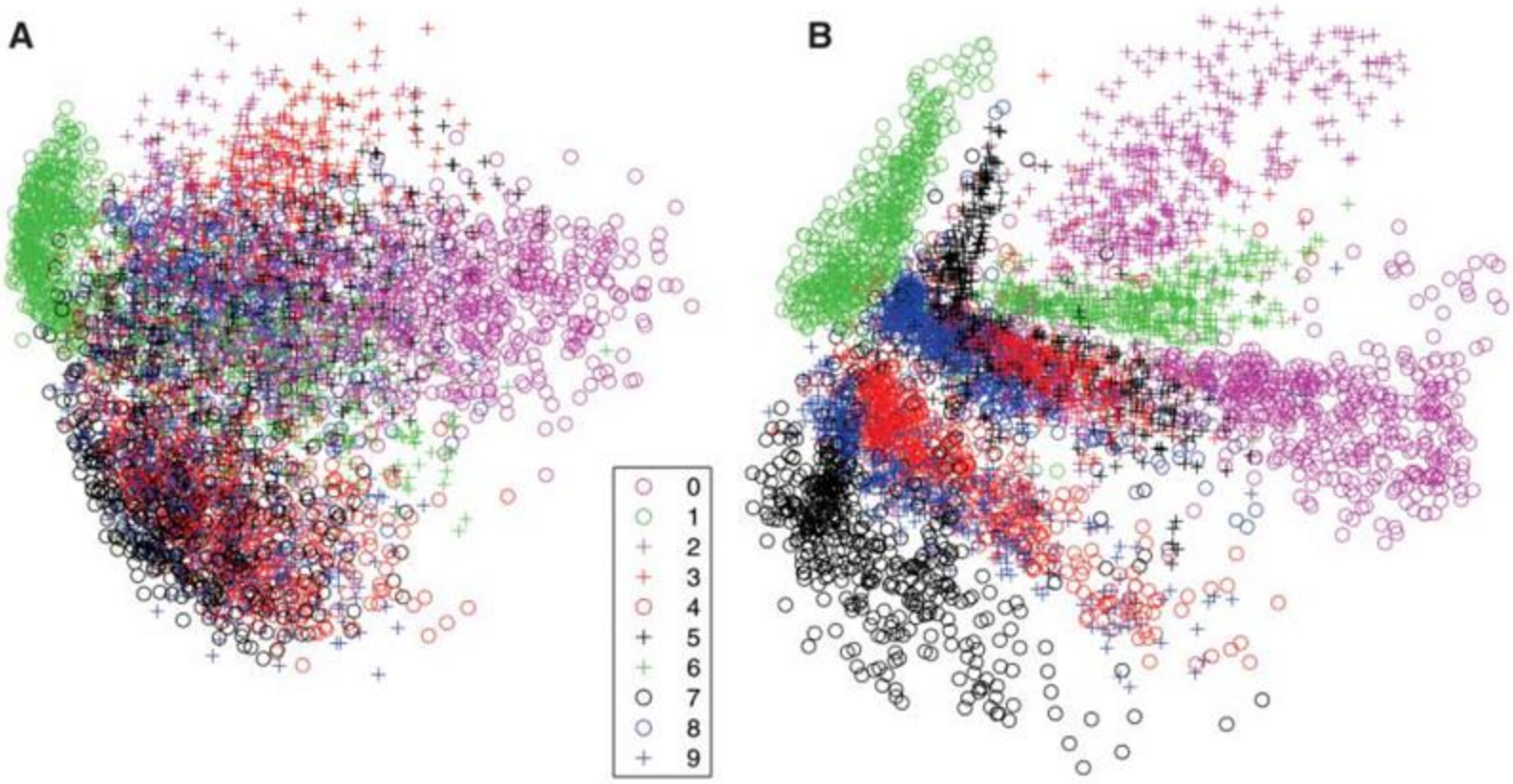
Dimensionality Reduction with Deep Learning

Hinton, Geoffrey E., and Ruslan R. Salakhutdinov.

"Reducing the dimensionality of data with neural networks."

Science 313.5786 (2006): 504-507.

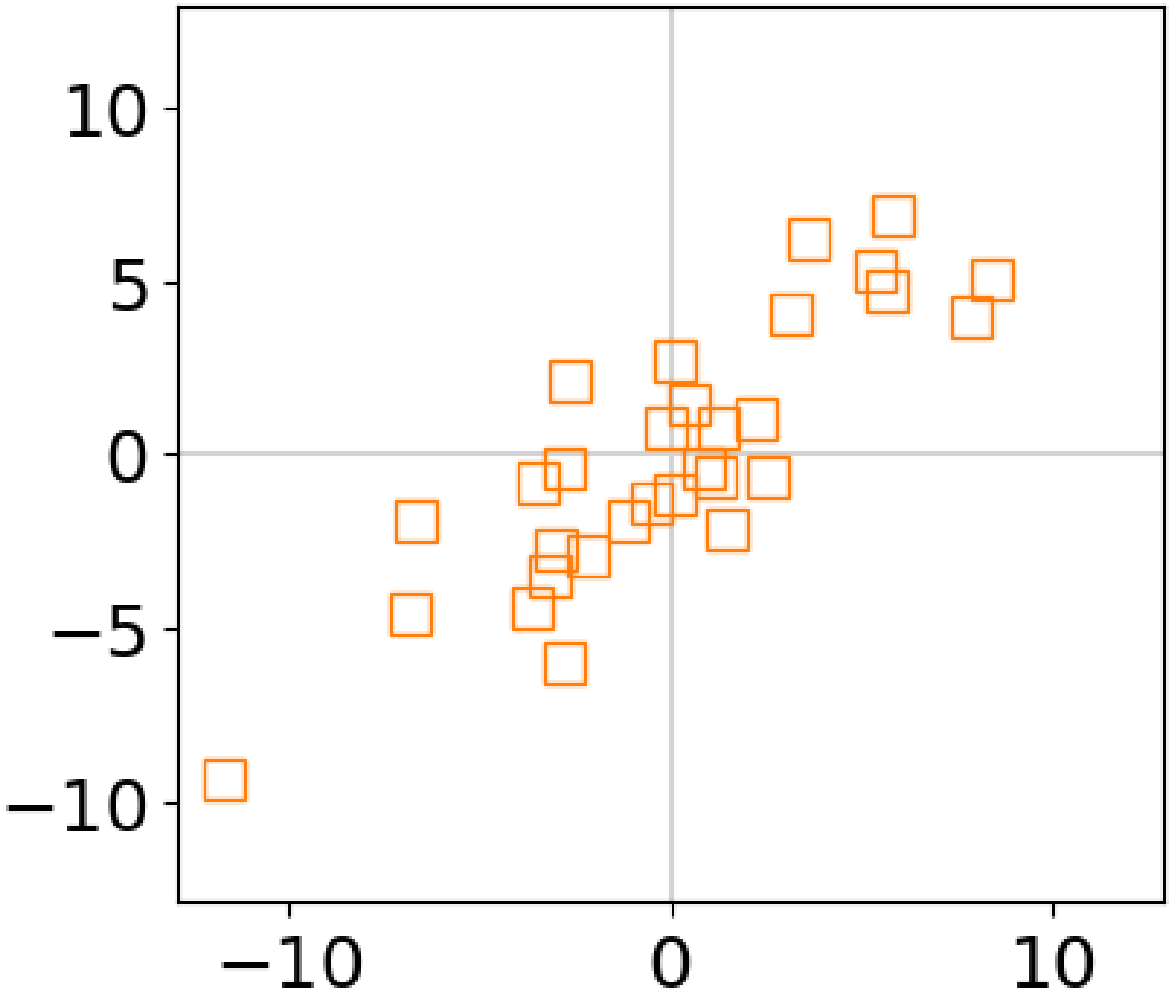
PCA



Autoencoder

Principle Component Axes

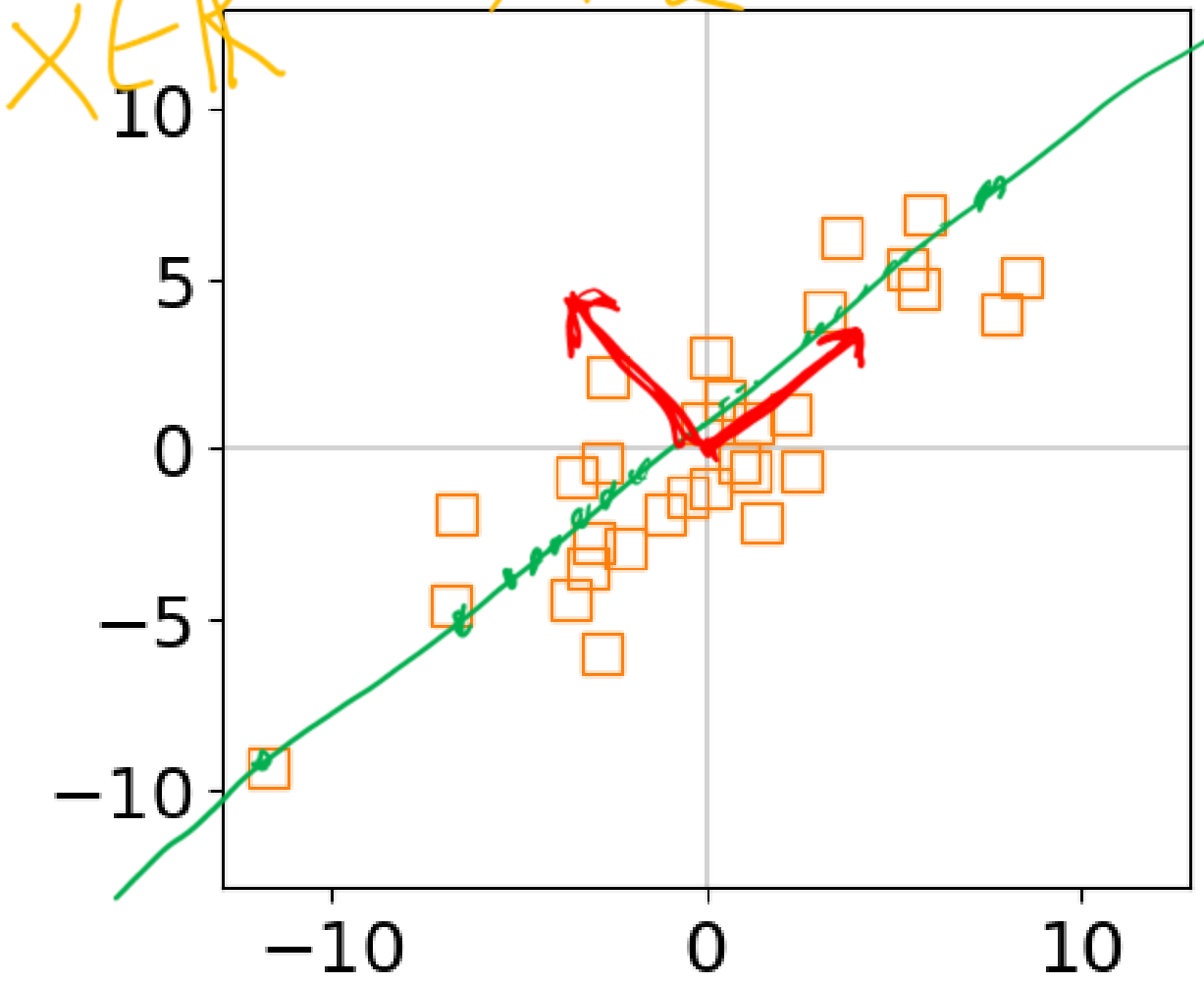
2-D Gaussian Data: 1st and 2nd principle component axes

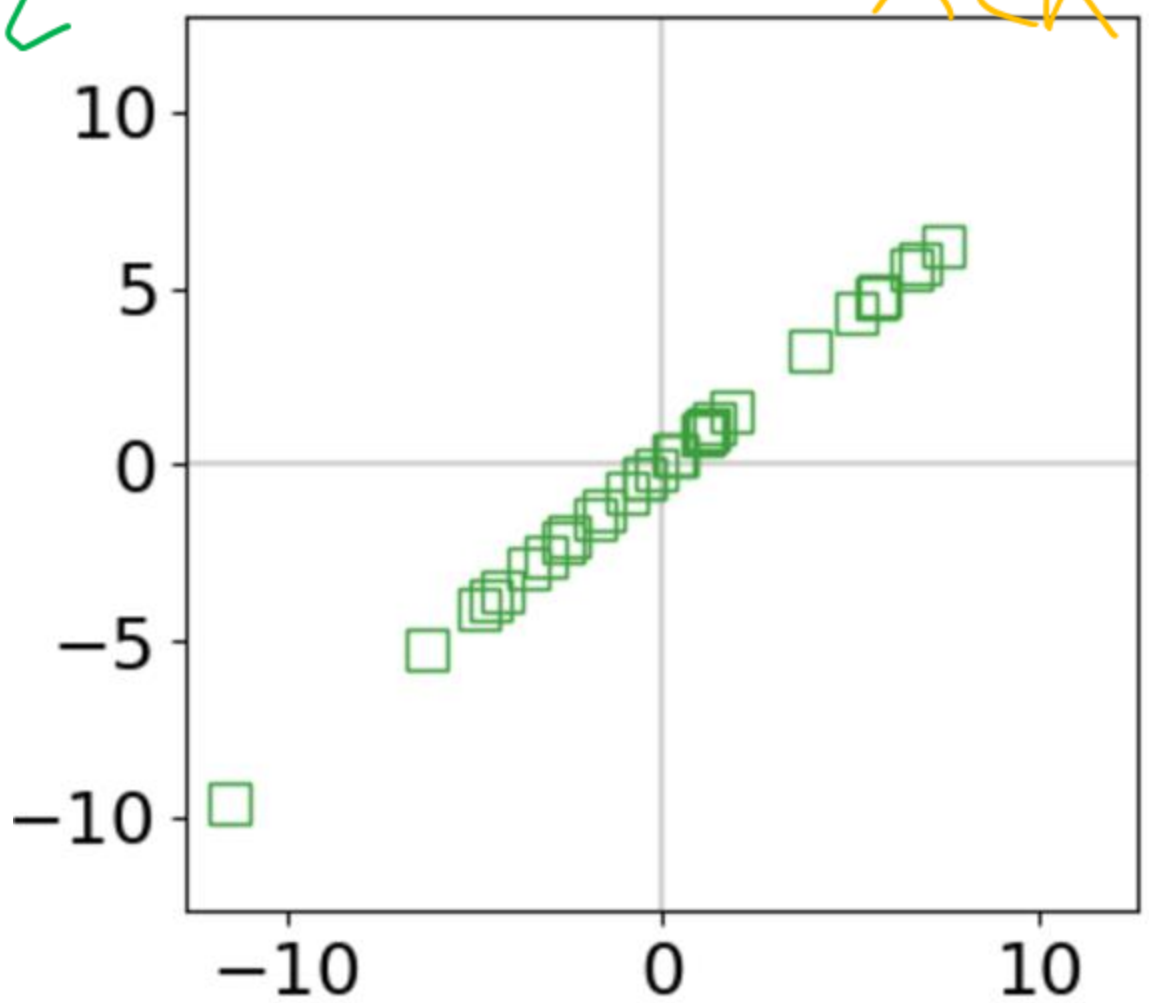


Spoiler: The PCA axes can be found using eigenvectors!

PCA Dimensionality Reduction

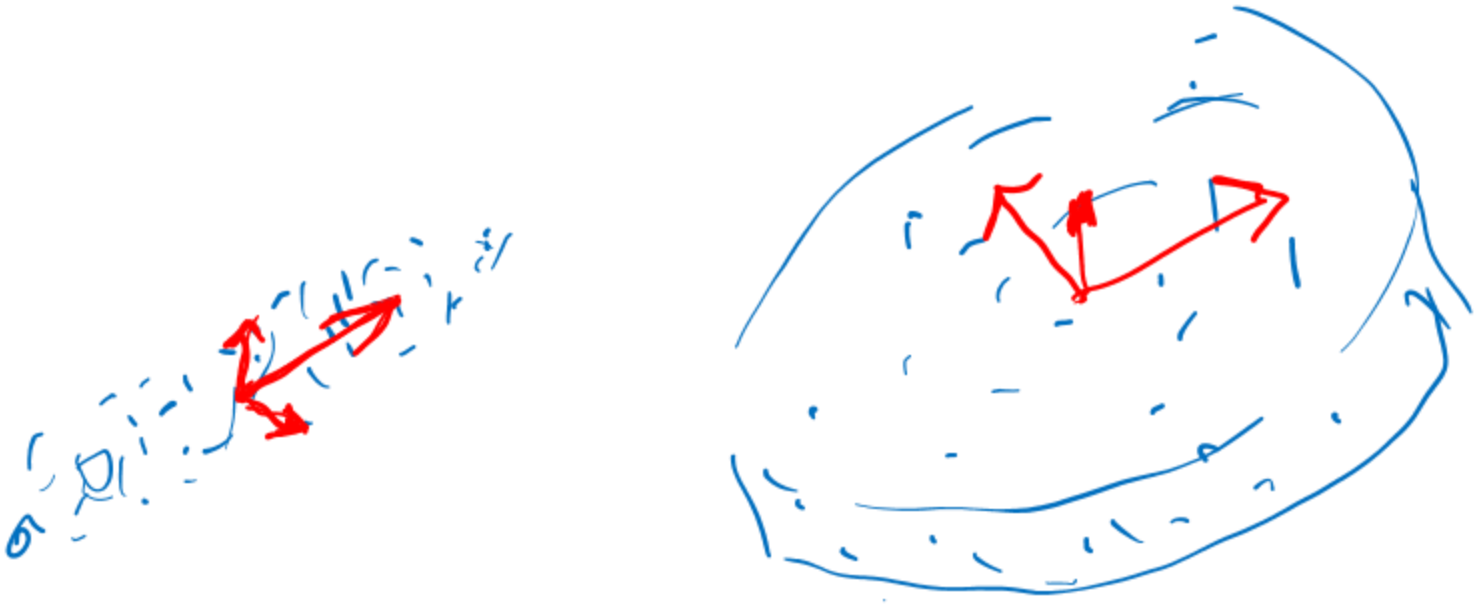
2-D Gaussian Data: Reduced along 1st principle component





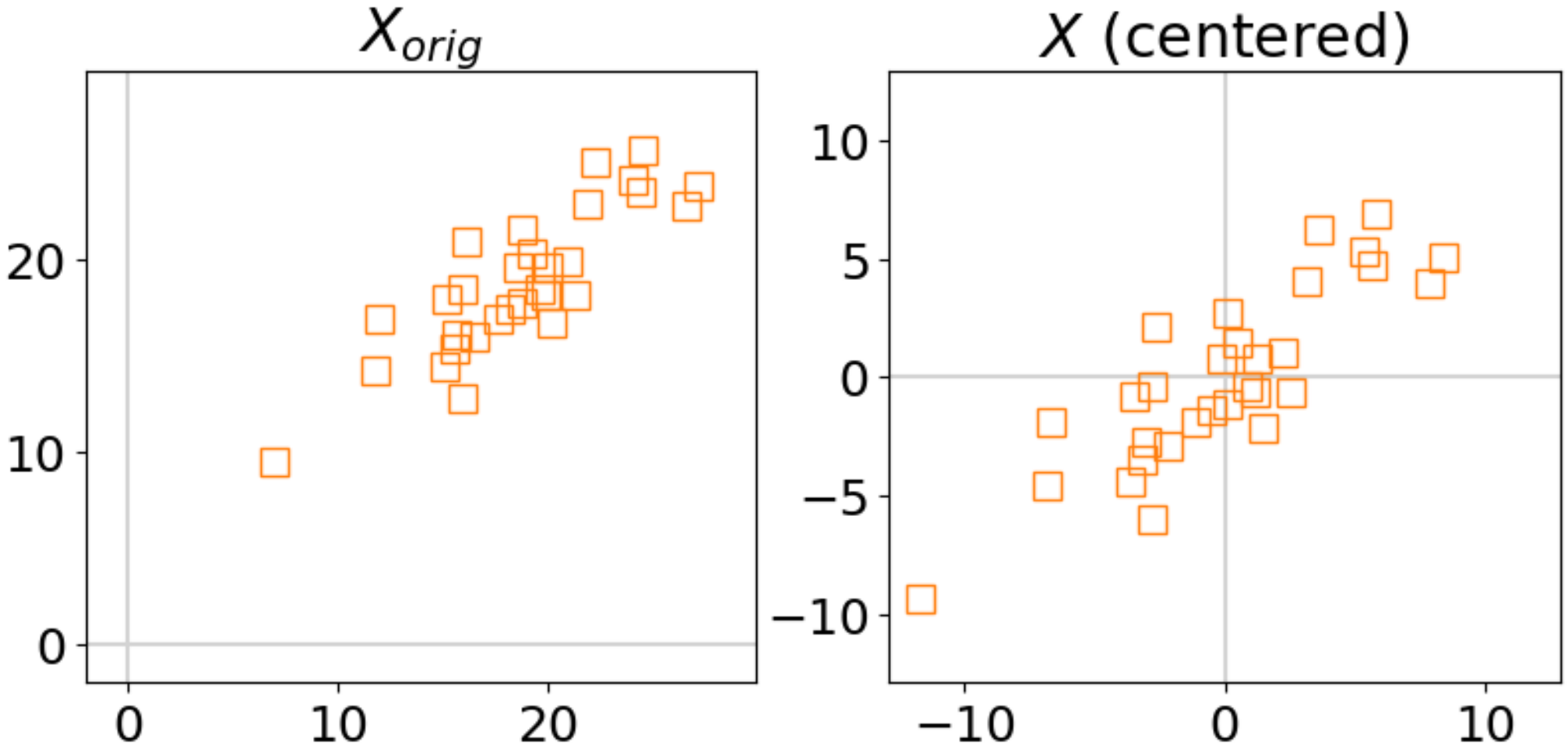
PCA Axes

3-D Data



PCA: Pre-processing

What if the data isn't centered

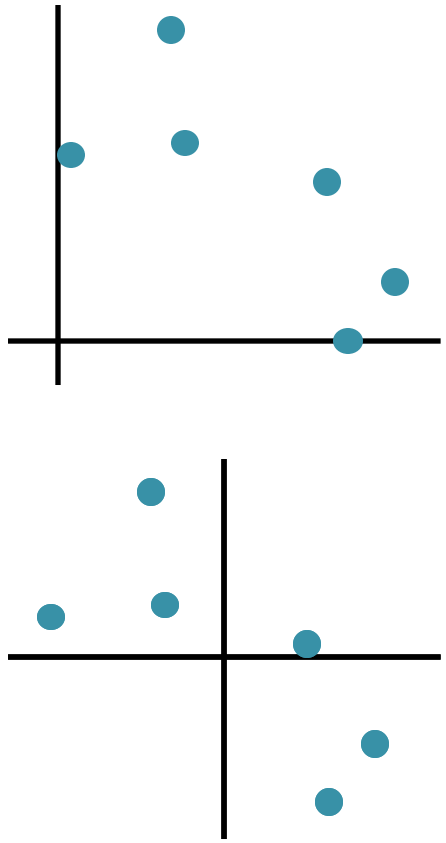


PCA: Centering Data

$$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N} \qquad \mathbf{X} = \begin{bmatrix} (\mathbf{x}^{(1)})^T \\ (\mathbf{x}^{(2)})^T \\ \vdots \\ (\mathbf{x}^{(N)})^T \end{bmatrix}$$

We assume the data is **centered**

$$\mu = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}^{(i)} = \mathbf{0}$$



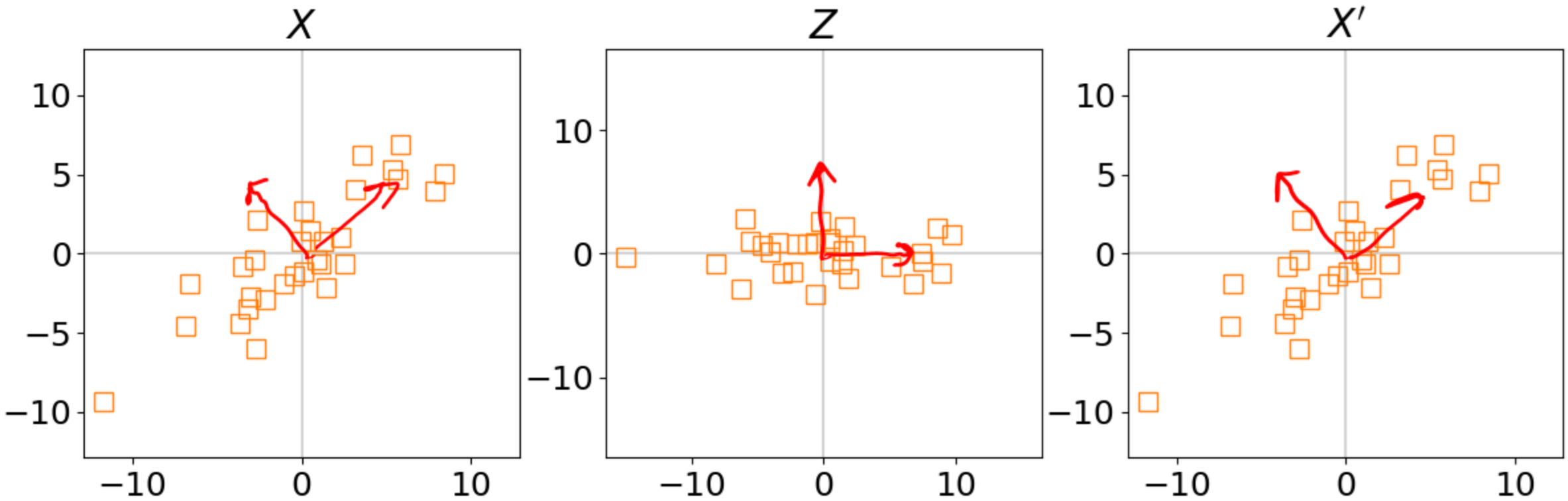
Q: What if your data is **not** centered?

A: Subtract off the sample mean

Slide from Matt Gormley

Rotation of Data (and back)

1. For any orthogonal matrix $V \in \mathbb{R}^{M \times M}$
2. Rotate to new space: $\mathbf{z}^{(i)} = V^\top \mathbf{x}^{(i)} \quad \forall i$
3. (Un)rotate back: $\mathbf{x}'^{(i)} = V\mathbf{z}^{(i)}$



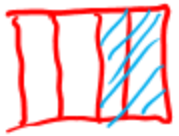
PCA Algorithm

PCA Algorithm

Input: X, X_{test}, K

$\lambda_1 \lambda_2 \lambda_3 \lambda_4$

1. Center data (and scale each axis) based on training data → X, X_{test}
2. V = eigenvectors($X^T X$) ← V_K =
3. Keep only the top K eigenvectors: V_K



$K=2$

eigenvectors in columns

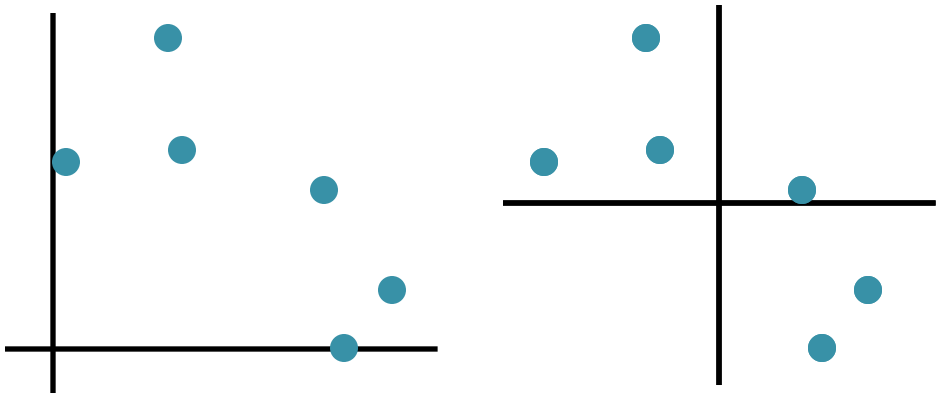
$\vec{z} = V^T \vec{x}$

$Z = XV$

PCA Algorithm

Input: X, X_{test}, K

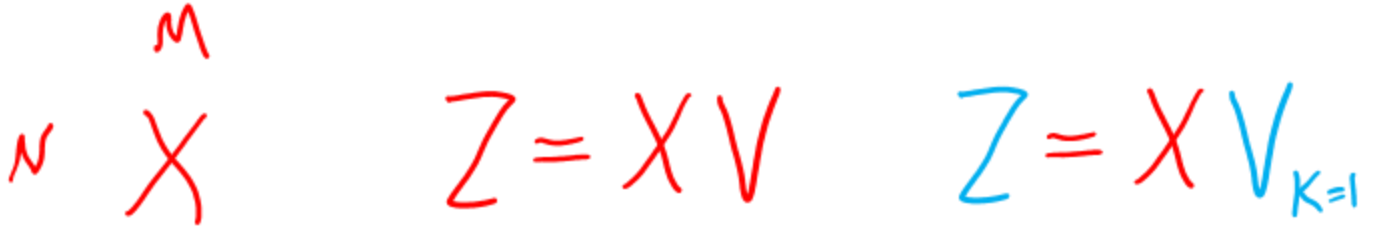
1. Center data (and scale each axis) based on training data → X, X_{test}
2. V = eigenvectors($X^T X$)
3. Keep only the top K eigenvectors: V_K

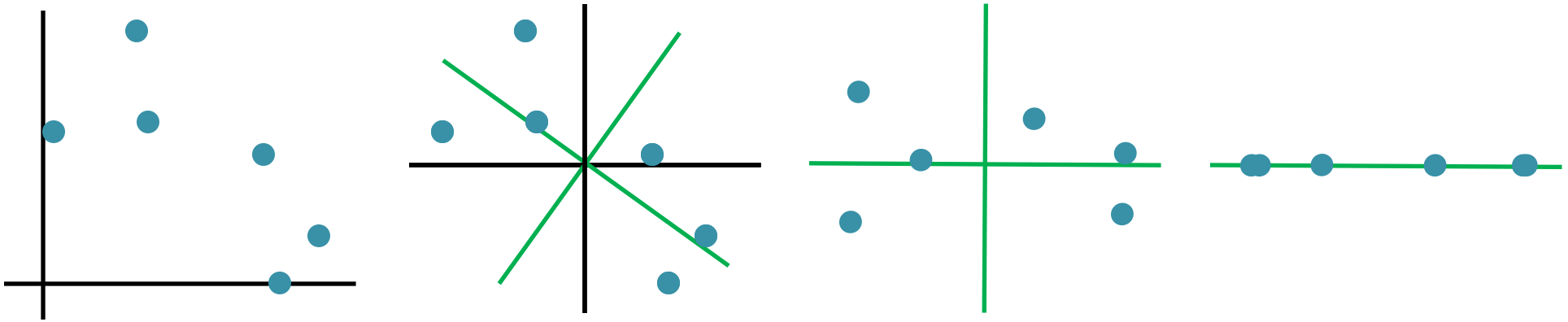


PCA Algorithm

Input: X, X_{test}, K

1. Center data (and scale each axis) based on training data → X, X_{test}
2. V = eigenvectors($X^T X$) ← V_K =  eigenvectors in columns
3. Keep only the top K eigenvectors: V_K (K=1)
4. $Z_{test} = X_{test} V_K$





PCA Algorithm

Input: X, X_{test}, K

1. Center data (and scale each axis) based on training data → X, X_{test}
2. V = eigenvectors($X^T X$)
3. Keep only the top K eigenvectors: V_K
4. $Z_{test} = X_{test} V_K$

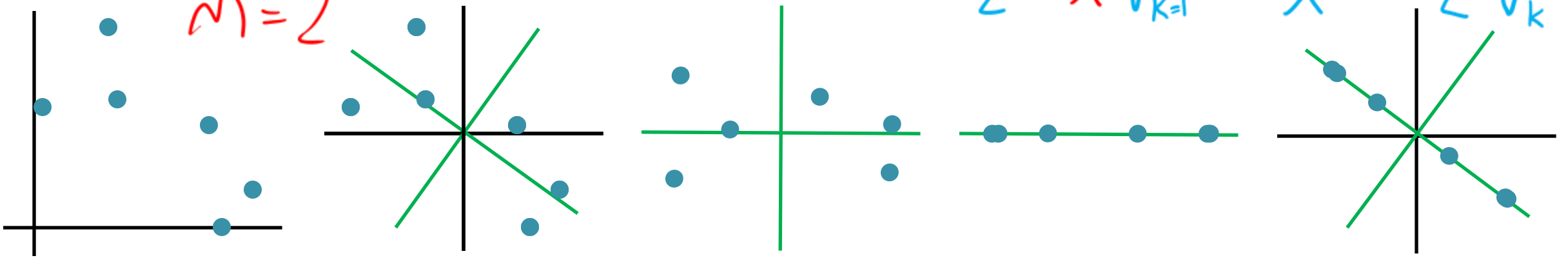
$K = 1$

Optionally, use V_K^T to rotate Z_{test} back to original subspace X'_{test} and uncenter

$M = 2$

$Z = X V_{K=1}$

$X' = Z V_k^T$

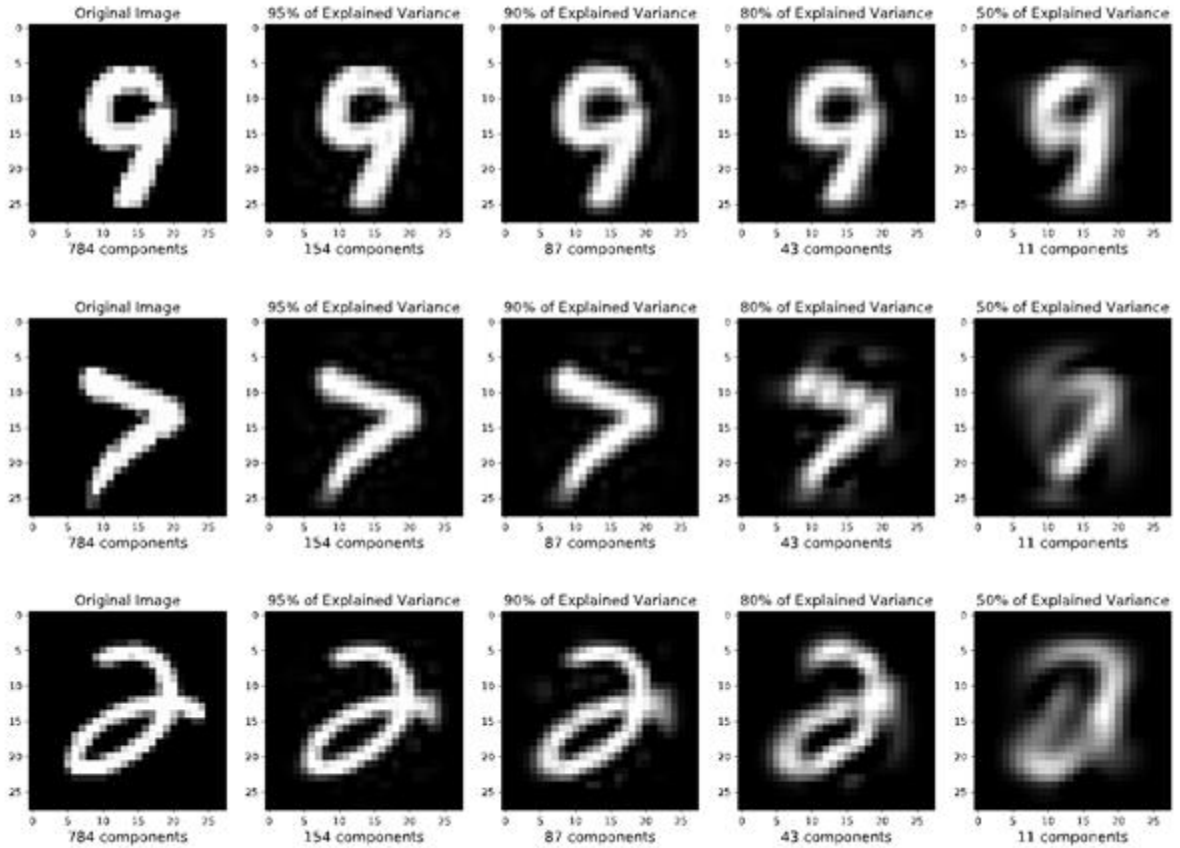


PCA Examples

Projecting MNIST digits

Task Setting

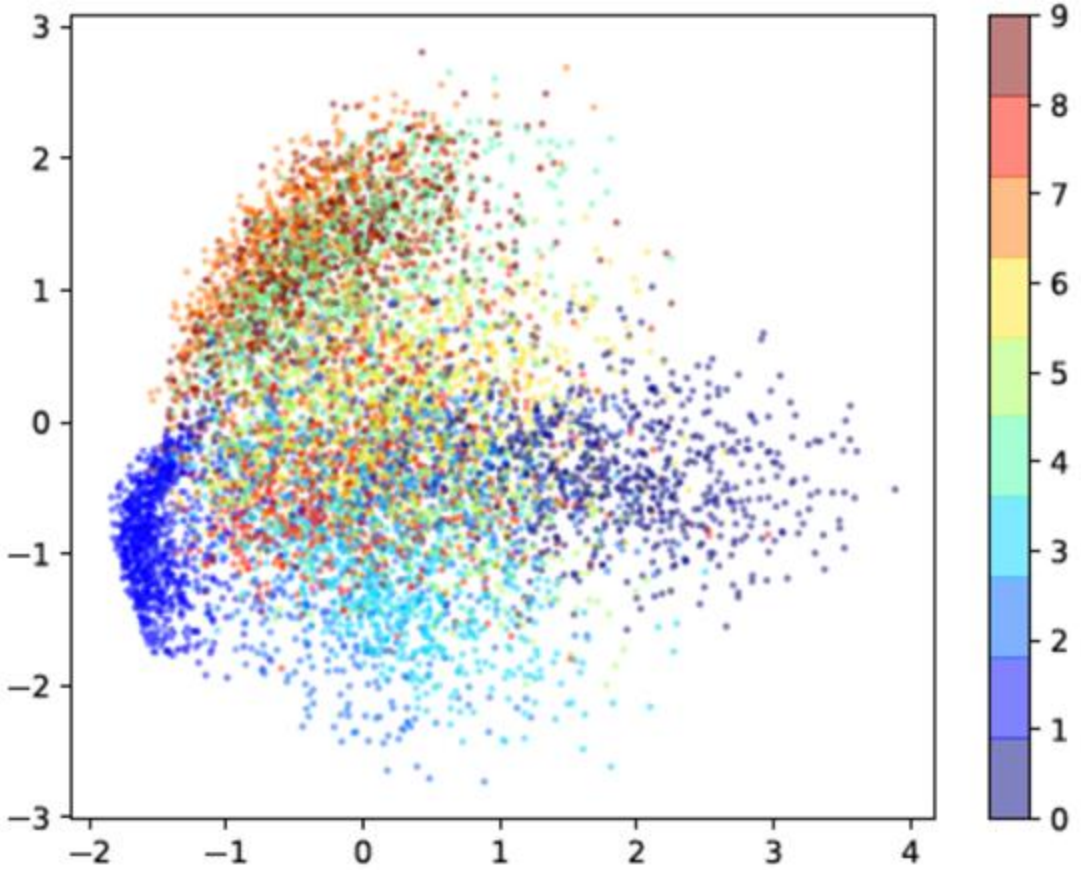
1. Take 28x28 images of digits and project them down to K components
2. Report percent of variance explained for K components
3. Then project back up to 28x28 image to visualize how much information was preserved



Projecting MNIST digits

Task Setting:

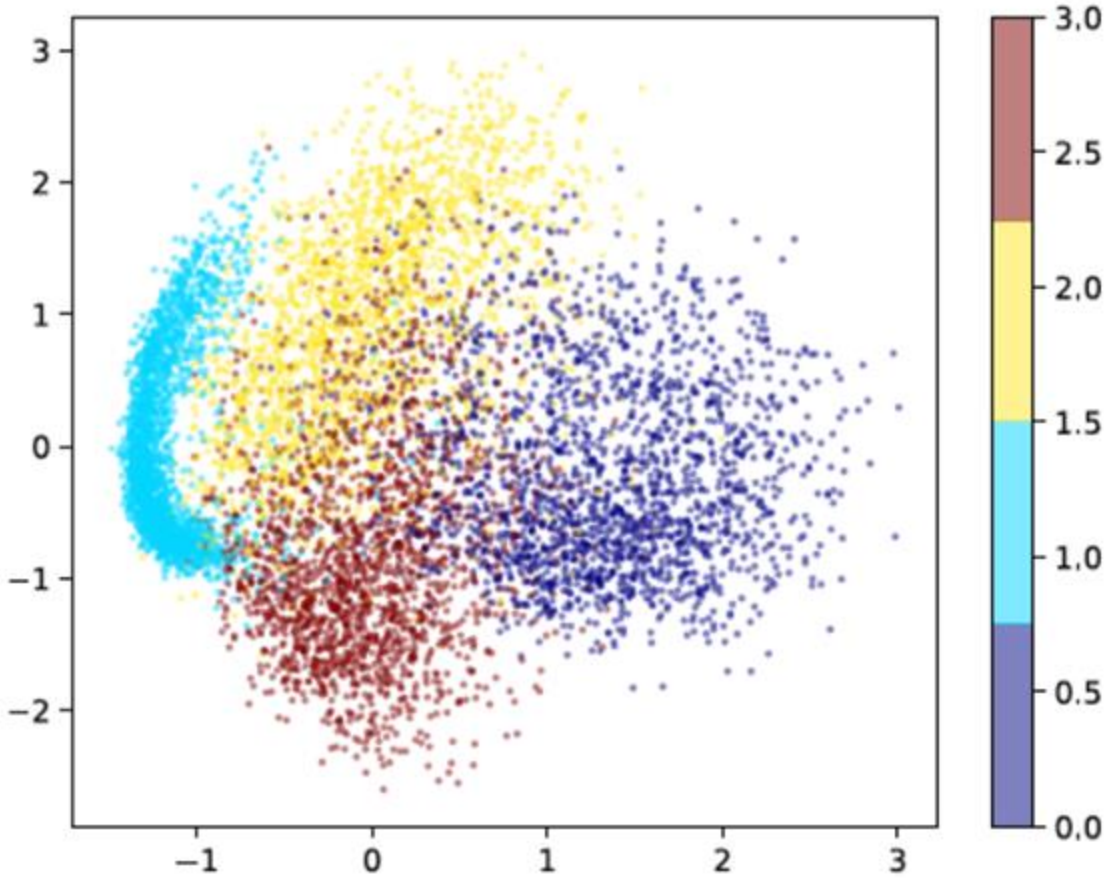
1. Take 28x28 images of digits and project them down to 2 components
2. Plot the 2 dimensional points



Projecting MNIST digits

Task Setting:

1. Take 28x28 images of digits and project them down to 2 components
2. Plot the 2 dimensional points



Growth Plate Imaging

Growth Plate Disruption and Limb Length Discrepancy

8 year-old boy with previous fracture
and 4cm leg length discrepancy

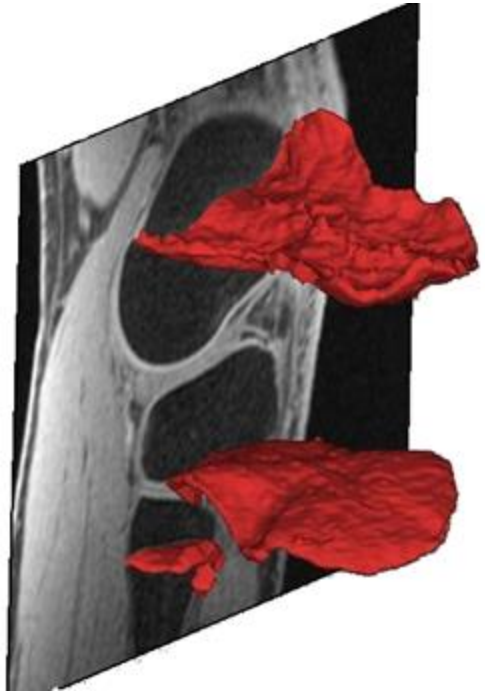
Images Courtesy
H. Potter, H.S.S.

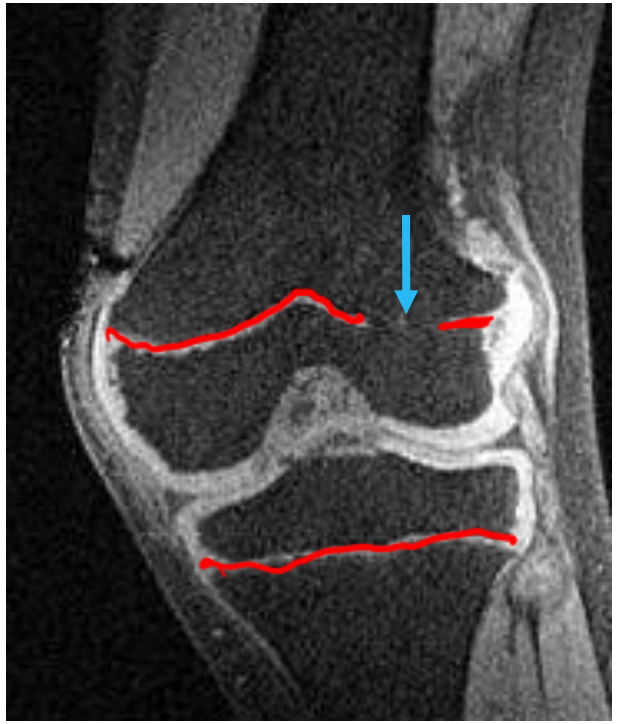
imagination at work

Growth Plate Imaging

Growth Plate Disruption and Limb Length Discrepancy

8 year-old boy with previous fracture
and 4cm leg length discrepancy

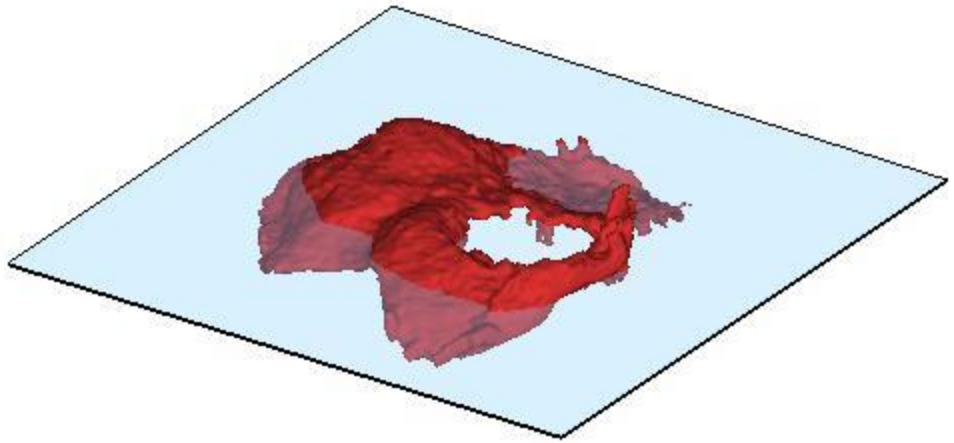




Images Courtesy
H. Potter, H.S.S.

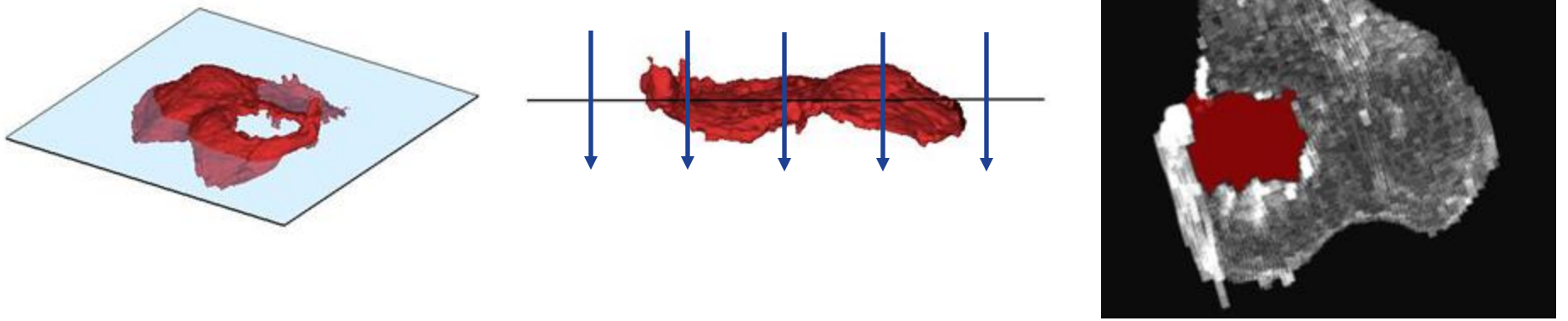
Growth Plate Imaging

Area Measurement



Growth Plate Imaging

Area Measurement



Flatten Growth Plate to Enable 2D Area Measurement

Outline

Unsupervised Learning

Dimensionality Reduction

Embedded Spaces and Feature Learning

Autoencoders

Principal Component Analysis (PCA)

- Examples: 2D and 3D
- PCA algorithm
- PCA, eigenvectors, and eigenvalues
- PCA objective and optimization

$$x' = W_2 W_1 x$$

$$x' = V_k (V_k)^T x$$

PCA Objective

Poll 1

What is the projection of point $\mathbf{x}$ onto vector $\mathbf{v}$, assuming that $\|\mathbf{v}\|_2 = 1$?

A. $\mathbf{vx}$

B. $\mathbf{v}^T\mathbf{x}$

C. $(\mathbf{v}^T\mathbf{x})\mathbf{v}$

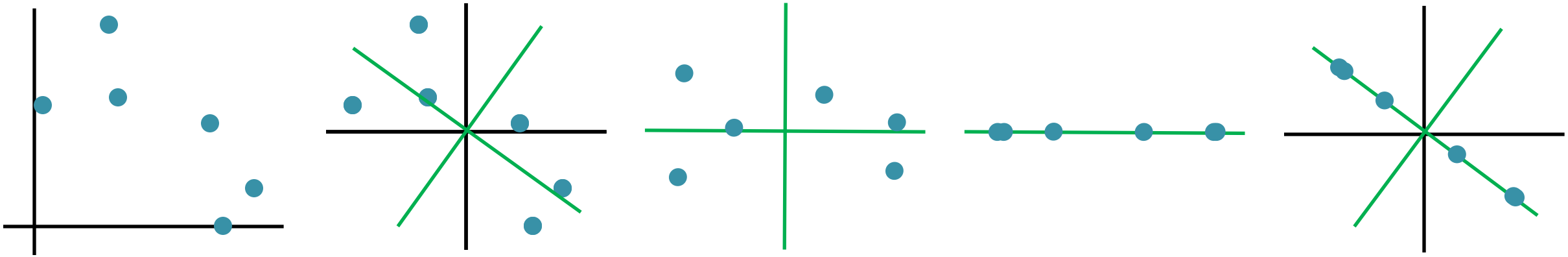
D. $\mathbf{v}^T\mathbf{x}\,\mathbf{x}^T\mathbf{v}$

PCA Algorithm

Input: X, X_{test}, K

1. Center data (and scale each axis) based on training data → X, X_{test}
2. V = eigenvectors($X^T X$)
3. Keep only the top K eigenvectors: V_K
4. $\mathrm{Z}_{test} = X_{test} V_K$

Optionally, use V_K^T to rotate Z_{test} back to original subspace $\mathrm{X'}_{test}$ and uncenter



Sketch of PCA

1. Select "best" $V \in \mathbb{R}^{M \times K}$
2. Project down: $\mathbf{z}^{(i)} = V^\top \mathbf{x}^{(i)} \quad \forall\, i$
3. Reconstruct up: $\mathbf{x}'^{(i)} = V\, \mathbf{z}^{(i)} \quad \forall\, i$

Select “Best” Vector

Reconstruction Error vs Variance of Projection

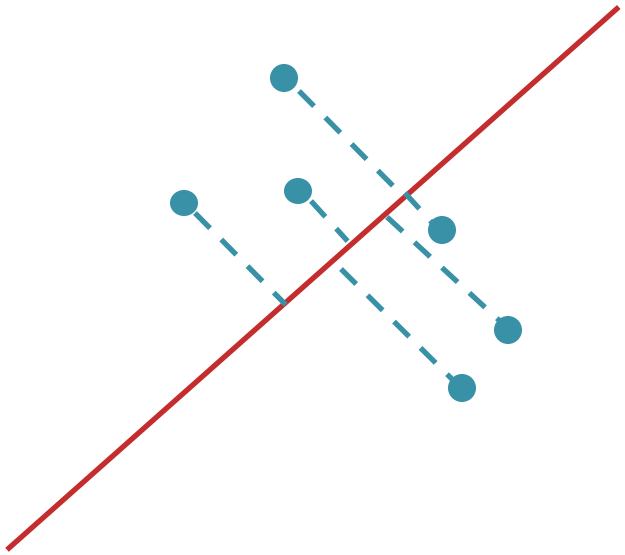
Poll 2 & Poll 3

Consider the two projections below

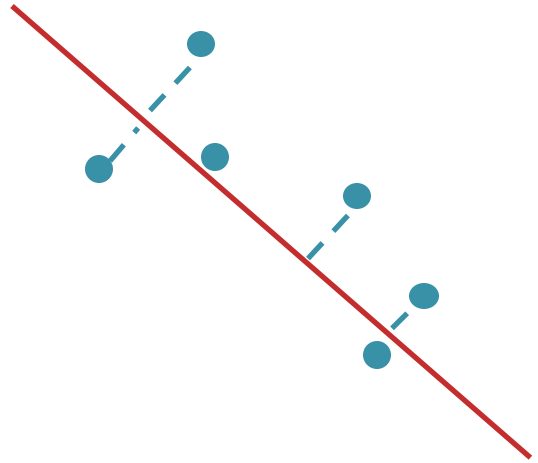
Poll 2: Which maximizes the variance?

Poll 3: Which minimizes the reconstruction error?

Option A

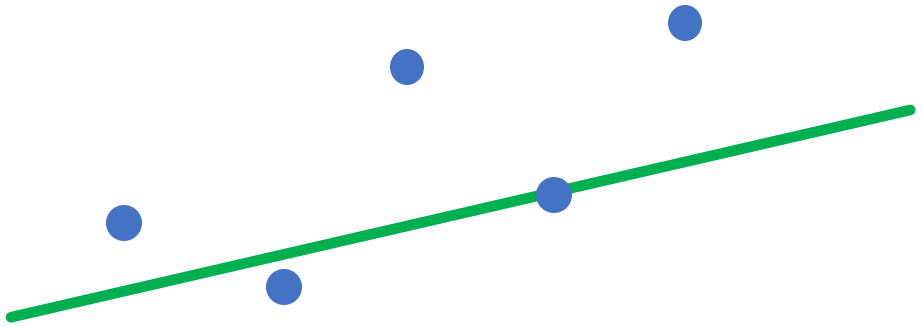


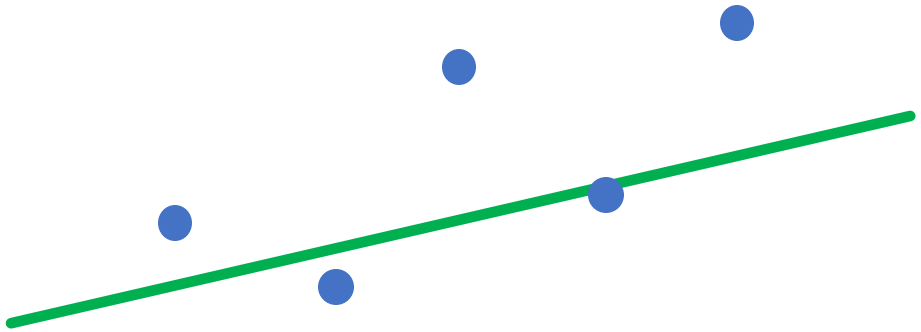
Option B



Select “Best” Vector

Reconstruction Error vs Variance of Projection





Reconstruction Error

$$\left\|\mathbf{x}^{(i)} - \mathbf{x}'^{(i)}\right\|_2^2$$

$$\mathbf{v}^* = \underset{\substack{\mathbf{v} \\ s.t. \|\mathbf{v}\|_2 = 1}}{\operatorname{argmin}} \sum_{i=1}^{N} \left\|\mathbf{x}^{(i)} - \left(\mathbf{v}^T \mathbf{x}^{(i)}\right)\mathbf{v}\right\|_2^2$$

Variance of Projection

$$\mathbf{v}^* = \underset{\substack{\mathbf{v} \\ s.t. \|\mathbf{v}\|_2 = 1}}{\operatorname{argmax}} \sum_{i=1}^{N} \left(\mathbf{v}^T \mathbf{x}^{(i)}\right)^2$$

PCA Objective Equivalence

Equivalence of Maximizing Variance and Minimizing Reconstruction Error

Claim: Minimizing the reconstruction error is equivalent to maximizing the variance.

Proof: First, note that:

$$||\mathbf{x}^{(i)} - (\mathbf{v}^T\mathbf{x}^{(i)})\mathbf{v}||^2 = ||\mathbf{x}^{(i)}||^2 - (\mathbf{v}^T\mathbf{x}^{(i)})^2 \tag{1}$$

since $\mathbf{v}^T\mathbf{v} = ||\mathbf{v}||^2 = 1$.

Substituting into the minimization problem, and removing the extraneous terms, we obtain the maximization problem.

$$\mathbf{v}^* = \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\text{argmin}} \frac{1}{N}\sum_{i=1}^{N} ||\mathbf{x}^{(i)} - (\mathbf{v}^T\mathbf{x}^{(i)})\mathbf{v}||^2 \tag{2}$$

$$= \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\text{argmin}} \frac{1}{N}\sum_{i=1}^{N} ||\mathbf{x}^{(i)}||^2 - (\mathbf{v}^T\mathbf{x}^{(i)})^2 \tag{3}$$

$$= \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\text{argmax}} \frac{1}{N}\sum_{i=1}^{N} (\mathbf{v}^T\mathbf{x}^{(i)})^2 \tag{4}$$

Sketch of PCA

1. Select "best" $V \in \mathbb{R}^{M \times K}$
2. Project down: $\mathbf{z}^{(i)} = V^\top \mathbf{x}^{(i)} \quad \forall\, i$
3. Reconstruct up: $\mathbf{x}'^{(i)} = V\, \mathbf{z}^{(i)} \quad \forall\, i$

Definition of PCA

1. Select $\mathbf{v}_1$ that best explains data
2. Select next $\mathbf{v}_j$ that
   i. Is orthogonal to $\mathbf{v}_1, \ldots, \mathbf{v}_{j-1}$
   ii. Best explains remaining data
3. Repeat 2 until desired amount of data is explained

PCA Eigenvalues and Eigenvectors

PCA: The First Principal Component

Use method of Lagrange multipliers to show that the first principle component is an eigenvalue of the covariance matrix

Poll 4

Given $X \in \mathbb{R}^{N \times M}$ with N M-dimensional datapoints, which is the covariance matrix?

A. $\frac{1}{N} X^\top X$

B. $\frac{1}{M} X^\top X$

C. $\frac{1}{\quad} X X^\top$

PCA: the First Principal Component

To find the first principal component, we wish to solve the following constrained optimization problem (variance maximization).

$$\mathbf{v}_1 = \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\operatorname{argmax}} \mathbf{v}^T \boldsymbol{\Sigma} \mathbf{v} \tag{1}$$

So we turn to the method of Lagrange multipliers. The Lagrangian is:

$$\mathcal{L}(\mathbf{v}, \lambda) = \mathbf{v}^T \boldsymbol{\Sigma} \mathbf{v} - \lambda(\mathbf{v}^T \mathbf{v} - 1) \tag{2}$$

Taking the derivative of the Lagrangian and setting to zero gives:

$$\frac{d}{d\mathbf{v}} \left(\mathbf{v}^T \boldsymbol{\Sigma} \mathbf{v} - \lambda(\mathbf{v}^T \mathbf{v} - 1) \right) = 0 \tag{3}$$

$$\boldsymbol{\Sigma} \mathbf{v} - \lambda \mathbf{v} = 0 \tag{4}$$

$$\boldsymbol{\Sigma} \mathbf{v} = \lambda \mathbf{v} \tag{5}$$

Recall: For a square matrix $\mathbf{A}$, the vector $\mathbf{v}$ is an **eigenvector** iff there exists **eigenvalue** λ such that:

$$\mathbf{A} \mathbf{v} = \lambda \mathbf{v} \tag{6}$$

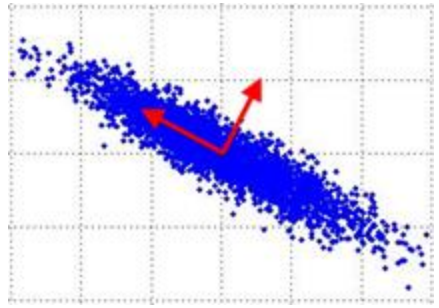
PCA: The Next Principal Component

Compute the next principal component from the residuals

Principal Component Analysis (PCA)

$(X^T X)v = \lambda v$, so v (the first PC) is the eigenvector of sample covariance matrix $X^T X$

Sample variance of projection $v^T X^T X v = \lambda v^T v = \lambda$



Thus, the eigenvalue λ denotes the amount of variability captured along that dimension (aka amount of energy along that dimension).

Eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots$

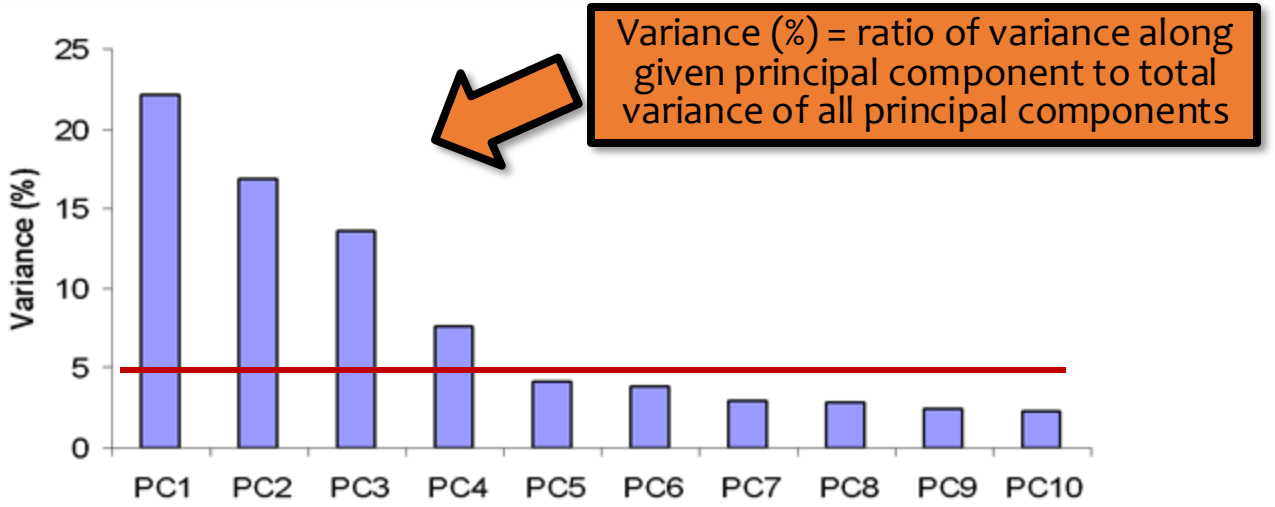
- The 1st PC v_1 is the eigenvector of the sample covariance matrix $X^T X$ associated with the largest eigenvalue
- The 2nd PC v_2 is the eigenvector of the sample covariance matrix $X^T X$ associated with the second largest eigenvalue
- And so on …

Slide from Nina Balcan

How Many PCs?

- For M original dimensions, sample covariance matrix is MxM, and has up to M eigenvectors. So M PCs.
- Where does dimensionality reduction come from?

  Can *ignore* the components of lesser significance.



- You do lose some information, but if the eigenvalues are small, you don't lose much
  - M dimensions in original data
  - calculate M eigenvectors and eigenvalues
  - choose only the first D eigenvectors, based on their eigenvalues
  - final data set has only D dimensions

SVD for PCA

SVD matrix factorization

$X = USV^T,\ A \in \mathbb{R}^{N \times M}$

U: $N \times N$ orthogonal matrix

- Columns of U are *left* singular vectors of X
- Columns of U are eigenvectors of XX^T

V: $M \times M$ orthogonal matrix

- Columns of V are *right* singular vectors of X
- Columns of V are eigenvectors of $X^T X$

S: $N \times M$ diagonal matrix

- Diagonal entries are singular values of X, σ_k
- Each σ_k^2 are the eigenvalues of both XX^T and $X^T X$!!

SVD for PCA

For any arbitrary matrix $\mathbf{A}$, SVD gives a decomposition:

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \tag{1}$$

where $\mathbf{\Lambda}$ is a diagonal matrix, and $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices.

Suppose we obtain an SVD of our data matrix $\mathbf{X}$, so that:

$$\mathbf{X} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \tag{1}$$

Now consider what happens when we rewrite $\mathbf{\Sigma} = \frac{1}{N}\mathbf{X}^T\mathbf{X}$ terms of this SVD.

$$\begin{aligned}
\mathbf{\Sigma} &= \frac{1}{N}\mathbf{X}^T\mathbf{X} && (2)\\
&= \frac{1}{N}(\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T)^T(\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T) && (3)\\
&= \frac{1}{N}(\mathbf{V}\mathbf{\Lambda}^T\mathbf{U}^T)(\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T) && (4)\\
&= \frac{1}{N}\mathbf{V}\mathbf{\Lambda}^T\mathbf{\Lambda}\mathbf{V}^T && (5)\\
&= \frac{1}{N}\mathbf{V}(\mathbf{\Lambda})^2\mathbf{V}^T && (6)
\end{aligned}$$

Above we used the fact that $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ since $\mathbf{U}$ is orthogonal by definition.

We find that $(\mathbf{\Lambda})^2$ is a diagonal matrix whose entries are $\Lambda_{ii} = \lambda_i^2$ the squares of the eigenvalues of the SVD of $\mathbf{X}$. Further, both $\mathbf{X}$ and $\mathbf{X}^T\mathbf{X}$ share the same eigenvectors in their SVD.

Thus, we can run SVD on $\mathbf{X}$ without ever instantiating the large $\mathbf{X}^T\mathbf{X}$ to obtain the necessary principal components more efficiently.