

Deep Learning: Principal Component Analysis

Lecture 05

Principal Component Analysis (PCA)

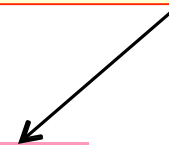
- A technique widely used for:

- dimensionality reduction.
- data compression.
- feature extraction.
- data visualization.

- Two equivalent definitions of PCA:

- 1) Project the data onto a lower dimensional space such that the variance of the projected data is *maximized*.
- 2) Project the data onto a lower dimensional space such that the mean squared distance between data points and their projections (average projection cost) is *minimized*.

maximum variance



minimum error

PCA (Maximum Variance)

- Let $X = \{\mathbf{x}_n\}_{1 \leq n \leq N}$ be a set of observations:
 - Each $\mathbf{x}_n \in \mathbb{R}^D$ (D is the dimensionality of \mathbf{x}_n).
- Project X onto an M dimensional space ($M < D$) such that the *variance* of the projected X is *maximized*.
 - Minimum error formulation leads to the same solution [PRML 12.1.2].
 - shows how PCA can be used for compression.
- Work out solution for $M = 1$, then generalize to any $M < D$.

PCA (Maximum Variance, $M = 1$)

- The lower dimensional space is defined by a vector $\mathbf{u}_1 \in \mathbb{R}^D$.
 - Only direction is important \Rightarrow choose $\|\mathbf{u}_1\|=1$.

- Each \mathbf{x}_n is projected onto a scalar $\mathbf{u}_1^T \mathbf{x}_n$

- The (sample) **mean** of the data is:

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{n=1}^N \mathbf{x}_n$$

- The (sample) **mean** of the projected data is $\mathbf{u}_1^T \bar{\mathbf{x}}$

PCA (Maximum Variance, $M = 1$)

- The (sample) **variance** of the projected data:

$$\frac{1}{N} \sum_{n=1}^N (\mathbf{u}_1^T \mathbf{x}_n - \mathbf{u}_1^T \bar{\mathbf{x}})^2 = \mathbf{u}_1^T \Sigma \mathbf{u}_1$$

where Σ is the **data covariance matrix**:

$$\Sigma = \frac{1}{N} \sum_{n=1}^N (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T$$

- Optimization problem is:

minimize:

$$-\mathbf{u}_1^T \sum \mathbf{x}_n$$

subject to:

$$\mathbf{u}_1^T \mathbf{u}_1 = 1$$

PCA (Maximum Variance, $M=1$)

- Lagrangian function:

$$L_P(\mathbf{u}_1, \lambda_1) = -\mathbf{u}_1^T \Sigma \mathbf{u}_1 + \lambda_1 (\mathbf{u}_1^T \mathbf{u}_1 - 1)$$

where λ_1 is the *Lagrangian multiplier* for constraint $\mathbf{u}_1^T \mathbf{u}_1 = 1$

- Solve:

$$\frac{\partial L_P}{\partial \mathbf{u}_1} = 0 \Rightarrow \Sigma \mathbf{u}_1 = \lambda_1 \mathbf{u}_1 \Rightarrow \begin{cases} \mathbf{u}_1 \text{ is an eigenvector of } \Sigma \\ \lambda_1 \text{ is an eigenvalue of } \Sigma \end{cases}$$

- $-\mathbf{u}_1^T \Sigma \mathbf{u}_1 = -\lambda_1 \mathbf{u}_1^T \mathbf{u}_1 = -\lambda_1$
- λ_1 is the largest eigenvalue of Σ .

PCA (Maximum Variance, $M=1$)

- λ_1 is the largest eigenvalue of Σ .
- \mathbf{u}_1 is the eigenvector corresponding to λ_1 :
 - also called the *first principal component*.
- For $M < D$ dimensions:
 - $\mathbf{u}_1 \mathbf{u}_2 \dots \mathbf{u}_M$ are the eigenvectors corresponding to the largest eigenvalues $\lambda_1 \lambda_2 \dots \lambda_M$ of Σ .
 - proof by induction.

PCA on Normalized Data

- Preprocess data $X = \{\mathbf{x}^{(i)}\}_{1 \leq i \leq m}$ such that:

- features have the same *mean* (0).
- features have the same *variance* (1).

1. Let $\mu = \frac{1}{m} \sum_{i=1}^m x^{(i)}.$

2. Replace each $x^{(i)}$ with $x^{(i)} - \mu.$

3. Let $\sigma_j^2 = \frac{1}{m} \sum_i (x_j^{(i)})^2$

4. Replace each $x_j^{(i)}$ with $x_j^{(i)} / \sigma_j.$

PCA on Natural Images

- **Stationarity**: the statistics in one part of the image should be the same as any other.
 - no need for variance normalization.
 - do mean normalization by subtracting from each image its mean intensity.

$$\mu^{(i)} := \frac{1}{n} \sum_{j=1}^n x_j^{(i)}$$
$$x_j^{(i)} := x_j^{(i)} - \mu^{(i)}$$

PCA on Normalized Data

- The covariance matrix is:

$$\Sigma = \frac{1}{m} XX^T = \frac{1}{m} \sum_{i=1}^m \mathbf{x}^{(i)} (\mathbf{x}^{(i)})^T$$

- The eigenvectors are:

$$\Sigma \mathbf{u}_j = \lambda_j \mathbf{u}_j \quad \text{where} \quad \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_D \quad \text{and} \quad \mathbf{u}_j^T \mathbf{u}_j = 1$$

- Equivalent with:

$$\Sigma U = U \Lambda$$

$$U = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_D] \quad \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_D \quad \text{and} \quad U^T U = I$$

$$\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_D)$$

PCA on Normalized Data

- U is an **orthogonal (rotation)** matrix, i.e. $U^T U = I$.
- The **full transformation (rotation)** of $x^{(i)}$ through PCA is:

$$y^{(i)} = U^T x^{(i)}$$

- $x^{(i)} = U y^{(i)}$

- The **k -dimensional projection** of $x^{(i)}$ through PCA is:

$$\hat{y}^{(i)} = U_{1,k}^T x^{(i)} = [u_1, \dots, u_k]^T x^{(i)}$$

- $\hat{x}^{(i)} = U_{1,k} \hat{y}^{(i)}$

- How many components k should be used?

How many components k should be used?

- Compute *percentage of variance retained* by $Y = \{y^{(i)}\}$, for each value of k :

$$\hat{y}^{(i)} = [u_1, \dots, u_k]^T x^{(i)}$$

$$\text{Var}(k) = \sum_{j=1}^k \text{Var}[\hat{y}_j] = \sum_{j=1}^k \text{Var}[u_j^T x]$$

$$= \sum_{j=1}^k \frac{1}{m} \sum_{i=1}^m \left(u_j^T x^{(i)} - u_j^T \bar{x} \right)^2 = \sum_{j=1}^k \frac{1}{m} \sum_{i=1}^m \underbrace{\left(u_j^T x^{(i)} \right)^2}_{\text{HW: Prove it is } \lambda_j} = \sum_{j=1}^k \lambda_j$$

HW: Prove it is λ_j

How many components k should be used?

- Compute *percentage of variance retained* by $Y = \{y^{(i)}\}$, for each value of k :

- Variance retained:

$$Var(k) = \sum_{j=1}^k \lambda_j$$

- Total variance:

$$Var(D) = \sum_{j=1}^D \lambda_j$$

- Percentage of variance retained: $P(k) = \frac{\sum_{j=1}^k \lambda_j}{\sum_{j=1}^D \lambda_j}$

How many components k should be used?

- Compute *percentage of variance retained* by $Y = \{y^{(i)}\}$, for each value of k :

$$P(k) = \frac{\sum_{j=1}^k \lambda_j}{\sum_{j=1}^D \lambda_j}$$

- Choose smallest k as to retain 99% of variance:

$$\hat{k} = \operatorname{argmin}_{1 \leq k \leq D} [P(k) \geq 0.99]$$

PCA as a Linear Auto-Encoder

- The **full transformation (rotation)** of $x^{(i)}$ through PCA is:

$$y = U^T x \Rightarrow x = Uy$$

- The **k -dimensional projection** of $x^{(i)}$ through PCA is:

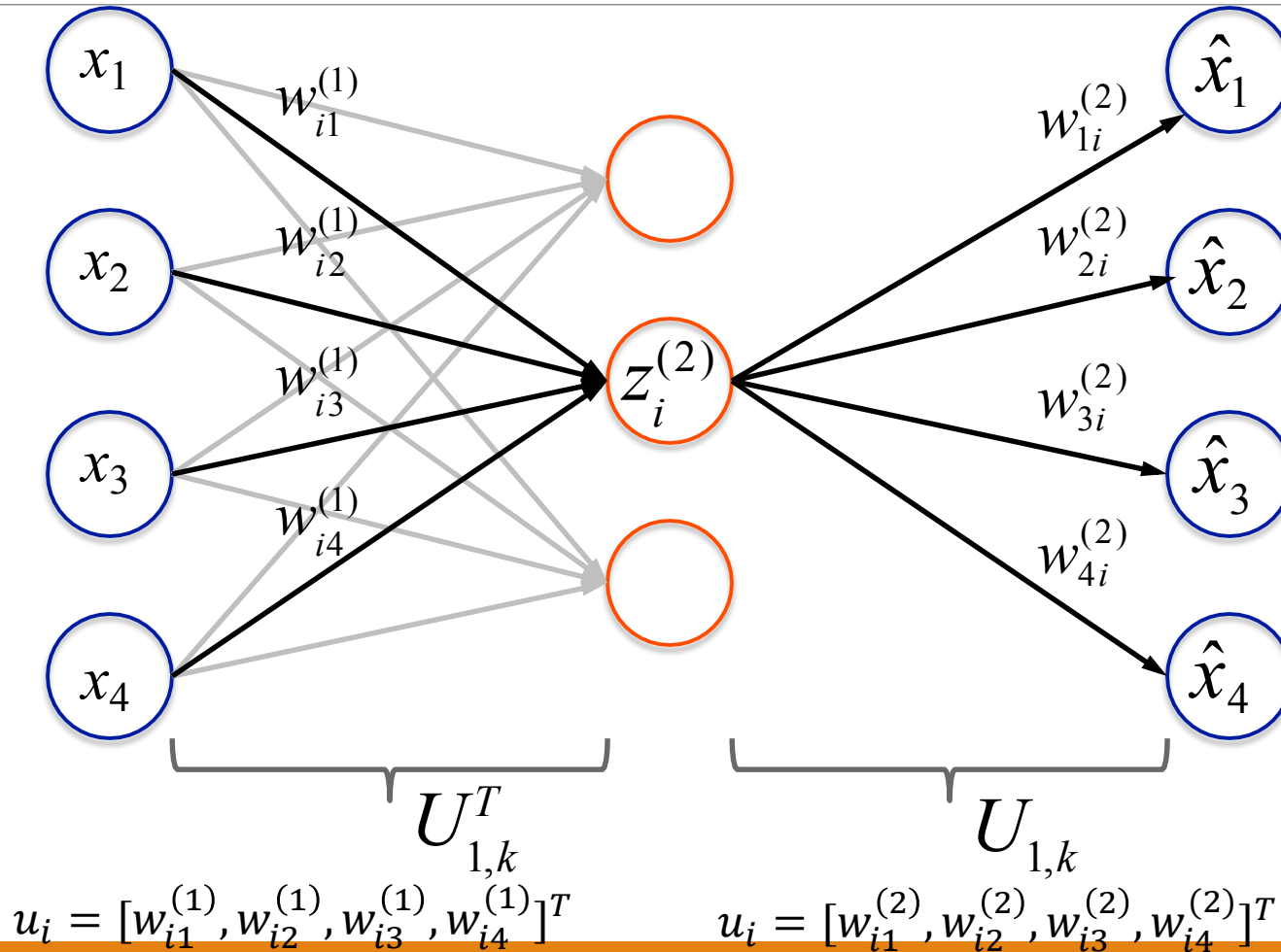
$$\hat{y} = U_{1,k}^T x = [u_1, \dots, u_k]^T x \Rightarrow \hat{x} = U_{1,k} \hat{y} = U_{1,k} U_{1,k}^T x$$

- The **minimum error** formulation of PCA:

$$U_{1,k}^* = \operatorname{argmin}_{U_{1,k}} \sum_{i=1}^m \left\| U_{1,k} U_{1,k}^T x^{(i)} - x^{(i)} \right\|^2$$

*a linear auto-encoder
with tied weights!*

PCA as a Linear Auto-Encoder



PCA and Decorrelation

- The **full transformation (rotation)** of $x^{(i)}$ through PCA is:

$$y^{(i)} = U^T x^{(i)} \Rightarrow Y = U^T X$$

- What is the covariance matrix of the rotated data Y?

$$\frac{1}{m} Y Y^T = \frac{1}{m} (U^T X) (U^T X)^T = \frac{1}{m} U^T X X^T U$$

$$= U^T \left(\frac{1}{m} X X^T \right) U = U^T \Sigma U = \Lambda$$

$$= \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_D)$$

**=> the features in y
are decorrelated!**

PCA Whitening (Sphering)

- The goal of **whitening** is to make the input *less redundant*, i.e. the learning algorithm sees a training input where:
 1. The features are **not correlated** with each other.

2. The features all have the **same variance**.

1. PCA already results in uncorrelated features:

$$y^{(i)} = U^T x^{(i)} \Leftrightarrow Y = U^T X \quad \frac{1}{m} Y Y^T = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_D)$$

2. Transform to identity covariance (**PCA Whitening**):

$$y_j^{(i)} = \frac{u_j^T x^{(i)}}{\sqrt{\lambda_j}} \Leftrightarrow y^{(i)} = \Lambda^{-1/2} U^T x^{(i)} \Leftrightarrow Y = \Lambda^{-1/2} U^T X$$

ZCA Whitening (Sphering)

- **Observation:** If Y has identity covariance and R is an orthogonal matrix, then RY has identity covariance.

1. PCA Whitening:

$$Y_{PCA} = \Lambda^{-1/2} U^T X$$

2. ZCA Whitening:

$$Y_{ZCA} = U Y_{PCA} = U \Lambda^{-1/2} U^T X$$

Out of all rotations, U makes Y_{ZCA} closest to original X .

Smoothing

- When eigenvalues λ_j are very close to 0, dividing by $\lambda_j^{-1/2}$ is numerically unstable.
- **Smoothing**: add a small ε to eigenvalues before scaling for PCA/ZCA whitening:

$$y_j^{(i)} = \frac{u^T x^{(i)}}{\sqrt{\lambda_j + \varepsilon}} \quad \varepsilon \approx 10^{-5}$$

- ZCA whitening is a rough model of how the biological eye (the retina) processes images (through retinal neurons).