Deep Learning: Principal Component Analysis

Lecture 03

Razvan C. Bunescu

School of Electrical Engineering and Computer Science

bunescu@ohio.edu
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.
Principal Component Analysis (PCA)
PCA (Maximum Variance)

• Let \( X = \{ x_n \}_{1 \leq n \leq N} \) be a set of observations:
  - Each \( x_n \in \mathbb{R}^D \) (\( D \) is the dimensionality of \( x_n \)).

• 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 $u_1 \in \mathbb{R}^D$.
  - Only direction is important $\Rightarrow$ choose $\|u_1\|=1$.
- Each $x_n$ is projected onto a scalar $u_1^T x_n$.
- The (sample) mean of the data is:
  $$\bar{x} = \frac{1}{N} \sum_{n=1}^{N} x_n$$
- The (sample) mean of the projected data is $u_1^T \bar{x}$.
PCA (Maximum Variance, $M = 1$)

- The (sample) variance of the projected data:

$$\frac{1}{N} \sum_{n=1}^{N} \left( u_1^T x_n - u_1^T \bar{x} \right)^2 = u_1^T \Sigma u_1$$

where $\Sigma$ is the data covariance matrix:

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

- Optimization problem is:

minimize:

$$-u_1^T \Sigma u_1$$

subject to:

$$u_1^T u_1 = 1$$
PCA (Maximum Variance, $M = 1$)

- Lagrangian function:

$$L_p(u_1, \lambda_1) = -u_1^T \Sigma u_1 + \lambda_1 (u_1^T u_1 - 1)$$

where $\lambda_1$ is the Lagrangian multiplier for constraint $u_1^T u_1 = 1$

- Solve:

$$\frac{\partial L_p}{\partial u_1} = 0 \implies \Sigma u_1 = \lambda_1 u_1 \implies \begin{cases} u_1 \text{ is an eigenvector of } \Sigma \\ \lambda_1 \text{ is an eigenvalue of } \Sigma \end{cases}$$

$$\implies -u_1^T \Sigma u_1 = -\lambda_1 u_1^T u_1 = -\lambda_1$$

$$\implies \lambda_1 \text{ is the largest eigenvalue of } \Sigma.$$
PCA (Maximum Variance, $M = 1$)

- $\lambda_1$ is the largest eigenvalue of $\Sigma$.
- $u_1$ is the eigenvector corresponding to $\lambda_1$:
  - also called the first principal component.

- For $M < D$ dimensions:
  - $u_1 \ u_2 \ldots \ u_M$ are the eigenvectors corresponding to the largest eigenvalues $\lambda_1 \ \lambda_2 \ldots \ \lambda_M$ of $\Sigma$.
  - proof by induction.
PCA on Normalized Data

- Preprocess data $X = \{x^{(i)}\}_{1 \leq i \leq m}$ 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 $\sigma_j^2 = \frac{1}{m} \sum_i (x^{(i)}_j)^2$

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

Lecture 3
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:

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

• The eigenvectors are:

\[
\Sigma u_j = \lambda_j u_j \quad \text{where} \quad \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_D \quad \text{and} \quad u_j^T u_j = 1
\]

• Equivalent with:

\[
\Sigma U = U \Lambda
\]

\[
U = [u_1, u_2, \ldots, u_D] \quad \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_D \quad \text{and} \quad U^T U = I
\]

\[
\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_D)
\]
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:

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

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

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

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

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

• How many components $k$ should be used?

Lecture 3
How many components $k$ should be used?

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

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

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

$$= \sum_{j=1}^{k} \frac{1}{m} \sum_{i=1}^{m} \left( u_j^T x^{(i)} - u_j^T \bar{x} \right)^2 = \sum_{j=1}^{k} \frac{1}{m} \sum_{i=1}^{m} (u_j^T x^{(i)})^2 = \sum_{j=1}^{k} \lambda_j$$

HW: Prove it is $\lambda_j$
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:
  $$\hat{k} = \arg\min_{1 \leq k \leq D} \left[ P(k) \geq 0.99 \right]$$
PCA on Normalized Data: $[x_1^{(i)}, x_2^{(i)}]^T$
Rotation through PCA: \[ [u_1^T x^{(i)}, u_2^T x^{(i)}]^T \]
1-Dimensional PCA Projection: $[u_1^T x^{(i)}, 0]^T$
1-Dimensional PCA Approximation: $u_1 u_1^T x^{(i)}$
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:

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

- The minimum error formulation of PCA:

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

*a linear auto-encoder with tied weights!*

Lecture 3
PCA as a Linear Auto-Encoder

\[
\begin{align*}
    \mathbf{u}_i &= \begin{bmatrix} w_{i1}^{(1)} & w_{i2}^{(1)} & w_{i3}^{(1)} & w_{i4}^{(1)} \end{bmatrix}^T \\
    \hat{\mathbf{x}}_i &= \begin{bmatrix} w_{1i}^{(2)} & w_{2i}^{(2)} & w_{3i}^{(2)} & w_{4i}^{(2)} \end{bmatrix}^T
\end{align*}
\]
PCA and Decorrelation

- The full transformation (rotation) of \( x^{(i)} \) through PCA is:
  \[
  y^{(i)} = U^T x^{(i)} \implies Y = U^T X
  \]

- What is the covariance matrix of the rotated data Y?
  \[
  \frac{1}{m} YY^T = \frac{1}{m} (U^T X)(U^T X)^T = \frac{1}{m} U^T XX^T U
  \]
  \[
  = U^T \left( \frac{1}{m} XX^T \right) U = U^T \Sigma U = \Lambda
  \]
  \[
  = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_D)
  \]
  
  \( \implies \) 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:
   \[ y^{(i)} = U^T x^{(i)} \iff Y = U^T X \]
   \[ \frac{1}{m} YY^T = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_D) \]

2. Transform to identity covariance (**PCA Whitening**):
   \[ y^{(i)}_j = \frac{u_j^T x^{(i)}}{\sqrt{\lambda_j}} \iff y^{(i)} = \Lambda^{-1/2} U^T x^{(i)} \iff Y = \Lambda^{-1/2} U^T X \]
Rotation through PCA: $[u_1^T x^{(i)}, u_2^T x^{(i)}]^T$
PCA Whitening: 
\[
\begin{bmatrix}
    \frac{u_1^T x^{(i)}}{\sqrt{\lambda_1}}, & \frac{u_2^T x^{(i)}}{\sqrt{\lambda_2}}
\end{bmatrix}^T
\]
ZCA Whitening (Sphering)

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

1. **PCA Whitening**:

   \[
   Y_{PCA} = \Lambda^{-\frac{1}{2}}U^TX
   \]

2. **ZCA Whitening**:

   \[
   Y_{ZCA} = UY_{PCA} = U\Lambda^{-\frac{1}{2}}U^TX
   \]

*Out of all rotations, \( U \) makes \( Y_{ZCA} \) closest to original \( X \).*
ZCA Whitening: $Y_{ZCA} = U \Lambda^{-\frac{1}{2}} U^T X$
Smoothing

• When eigenvalues $\lambda_j$ are very close to 0, dividing by $\lambda_j^{-1/2}$ is numerically unstable.

• **Smoothing**: add a small $\varepsilon$ to eigenvalues before scaling for PCA/ZCA whitening:

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

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