CSC2515 Tutorial: Dimensionality Reduction

 Presented by Ali Punjani
 Slides borrowed from Kevin Swersky, Ruslan Salakhutdinov, Laurent Charlin
Dimensionality Reduction

- We have some data \( X \in \mathbb{R}^{N \times D} \).
- \( D \) may be huge, etc.
- We would like to find a new representation where \( K \ll D \).
  - For computational reasons.
  - To better understand (e.g., visualize) the data.
  - For compression.
  - ...
- We will restrict ourselves to linear transformations for the time being.
Example

- In this dataset, there are only 3 degrees of freedom: horizontal and vertical translations, and rotations.

- Yet each image contains 784 pixels, so $X$ will be 784 elements wide.
Abstract Visualization
What is a Good Transformation?

- Goal is to find good directions $u$ that preserves “important” aspects of the data.

- In a linear setting: $z = x^T u$

- This will turn out to be the top-$K$ eigenvalues of the data covariance.

- Two ways to view this:
  1. Find directions of *maximum variation*
  2. Find projections that *minimize reconstruction error*
Principal Component Analysis (Maximum Variance)

\[
\text{maximize} \quad \frac{1}{2N} \sum_{n=1}^{N} (u_1^T x_n - u_1^T \bar{x}_n)^2 = u_1^T S u_1
\]

i.e., variance of the projected data

where the sample mean and covariance are given by:

\[
\bar{x} = \frac{1}{N} \sum_{n=1}^{N} x_n
\]

\[
S = \frac{1}{N} \sum_{n=1}^{N} (x_n - \bar{x})(x_n - \bar{x})^T
\]
Finding $u_1$

- We want to maximize $u_1^T Su_1$

subject to $||u_1|| = 1$

(since we are finding a direction)

- Use lagrange multiplier $\alpha_1$ to express this as

$$u_1^T Su_1 + \alpha_1 (1 - u_1^T u_1)$$
Finding $u_1$

- Take derivative and set to 0

\[ Su_1 - \alpha_1 u_1 = 0 \]

\[ Su_1 = \alpha_1 u_1 \]

- So $u_1$ is an eigenvector of $S$ with eigenvalue $\alpha_1$

- In fact it must be the eigenvector with maximum eigenvalue, since this minimizes the objective.
Finding $u_2$

maximize  $u_2^T Su_2$

subject to  $||u_2|| = 1$

$u_2^T u_1 = 0$

Lagrange form:

$u_2^T Su_2 + \alpha_2 (1 - u_2^T u_2) - \beta u_2^T u_1$

Finding $\beta$:

$\frac{\partial}{\partial u_2} = Su_2 - \alpha_2 u_2 - \beta u_1 = 0$

$\implies u_1^T Su_2 - \alpha_2 u_1^T u_2 - \beta u_1^T u_1 = 0$

$\implies \alpha_1 u_1^T u_2 - \alpha_2 u_1^T u_2 - \beta u_1^T u_1 = 0$

$\implies \alpha_1 \cdot 0 - \alpha_2 \cdot 0 - \beta \cdot 1 = 0$

$\implies \beta = 0$
Finding $u_2$

Maximize $u_2^T S u_2$

Subject to $\|u_2\| = 1$

$u_2^T u_1 = 0$

Lagrange form:

$$u_2^T S u_2 + \alpha_2 (1 - u_2^T u_2) - \beta u_2^T u_1$$

Finding $\alpha_2$:

$$\frac{\partial}{\partial u_2} (Su_2 - \alpha_2 u_2) = 0$$

$$\implies Su_2 = \alpha_2 u_2$$

So $\alpha_2$ must be the second largest eigenvector of $S$. 
PCA in General

- We can compute the entire PCA solution by just computing the eigenvectors with the top-k eigenvalues.
- These can be found using the singular value decomposition of S.
How do we choose the number of components?

\[
\sum_{i=1}^{M} \alpha_i \over \sum_{i=1}^{N} \alpha_i
\]

- Look at the spectrum of covariance, pick K to capture most of the variation.
- More principled: Bayesian treatment (beyond this course).
Demo

- Eigenfaces
Principal Component Analysis (Minimum Reconstruction Error)

- We can also think of PCA as minimizing the reconstruction error of the compressed data.

\[
\text{minimize} = \frac{1}{2N} \sum_{n=1}^{N} ||x_n - \hat{x}_n||^2
\]

- We will omit the details for now, but the key is that we define some K-dimensional basis such that:

\[
\hat{x} = Wx + \text{const}
\]

- The solution will turn out to be the same as the minimum variance formulation.
Reconstruction

- PCA learns to represent vectors in terms of sums of basis vectors.
- For images, e.g.,

\[ \text{image} = a1 + a2 + a3 + \ldots + a100 + \ldots \]
PCA for Compression

D = 1  D = 5  D = 10

D = 50  D = 100  D = 200

321x481 image, D is the number of basis vectors used

D in this slide is the same as K in the previous slides
Relation to Neural Networks

- An autoencoder is a neural network whose outputs are its own inputs. The goal is to minimize reconstruction error.
Autoencoders

- Define:
  \[ z = g(Wx) \]
  \[ \hat{x} = g(Vz) \]

- Goal:
  \[ \text{minimize} \frac{1}{2N} \sum_{n=1}^{N} ||x_n - \hat{x}_n||^2 \]

- If \( g \) is linear:
  \[ \text{minimize} \frac{1}{2N} \sum_{n=1}^{N} ||x_n - VWx_n||^2 \]

- In other words, the optimal solution is PCA.
Autoencoders

- What if $g$ is not linear?
- Then we are basically doing *nonlinear* PCA.
- Some subtleties (see Bishop) but in general you can take the above statement as fact.
<table>
<thead>
<tr>
<th>Real data</th>
</tr>
</thead>
<tbody>
<tr>
<td>30-d deep autoencoder</td>
</tr>
<tr>
<td>30-d logistic PCA</td>
</tr>
<tr>
<td>30-d PCA</td>
</tr>
</tbody>
</table>
Autoencoder 2D Topic Space

Legal/Judicial

Inflation

Energy Markets

Disasters and Accidents

Interbank Markets

Government Borrowings

European Community Monetary/ Economic Accounts/ Earnings

Reuter Corp.: Linear mapping 2-D topic space

Chinese Economic Leading Indicators

EuroCornely

German Earnings

Law Journals

Economy

Economists

Economist